On the Approximability of Range Assignment Problems

Inaugural-Dissertation
zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultät
der Universität zu Köln

vorgelegt von
Bernhard Fuchs
aus Leverkusen

Köln 2007

Berichterstatter: Prof. Dr. R. Schrader
Prof. Dr. U. Faigle
Dr. W. Kern

Zusammenfassung

Diese Arbeit behandelt Fragestellungen aus dem Bereich der kombinatorischen Optimierung, die sich bei der Datenübertragung in Funknetzen ergeben. Als Probleminstanz haben wir eine Menge von Sende- und Empfangsstationen mit ihren jeweiligen Entfernungen gegeben und sollen eine bestimmte Netzwerkeigenschaft herstellen, wobei die Gesamtenergieaufnahme des Netzwerks möglichst gering gehalten werden soll. Der Senderadius jeder einzelnen Station kann dabei individuell eingestellt werden. Die Lösung eines solchen Problems ist also eine Zuordnung von Senderadien zu den Stationen, die die geforderte Netzwerkeigenschaft mit dem niedrigst möglichem Gesamtenergiebedarf herstellt.

Wir untersuchen in dieser Arbeit drei grundlegende Problemstellungen dieses Typs. Zunächst lassen wir auch sehr abstrakte Distanzfunktion zu, die nicht geometrischen Ursprungs sind. Wir analysieren die effiziente genaue und näherungsweise Lösbarkeit der einzelnen Probleme für verschiedene Arten von möglichen Distanzfunktionen. Unsere Untersuchungen zeigen auch wichtige Unterschiede der drei Problemstellungen auf, die nicht ohne weiteres augenscheinlich sind.

Danach beschäftigen wir uns mit geometrischen Instanzen, die in der Forschung hinsichtlich Komplexität und Approximierbarkeit bereits etabliert sind. Unser Beitrag hierzu sind neue Reduktionen für die betrachteten Probleme, die sich als einfacher und flexibler als die bisher gebrauchten erweisen und somit neue und verbesserte Ergebnisse liefern. Offene Fragen aus früheren Arbeiten konnten mit unseren Reduktionen beantwortet werden.

Wir behandeln auch Approximationsalgorithmen für diese Probleme. Wir untersuchen einen bereits bekannten Algorithmus hinsichtlich seiner Gütegarantie in Abhängigkeit von der Größe der Probleminstanz. Danach geben wir eine detaillierte Analyse zweier natürlicher „Greedy“-Algorithmen. Abschließend entwickeln wir einen neuen Approximationsalgorithmus für eines der drei Probleme, und geben ein Approximationsschema für spezielle geometrische Instanzen an.

Abstract

We consider combinatorial optimization problems motivated by the following scenario. We are given a set of radio stations which can all send and receive data via wireless communication. Each radio station can be assigned an individual range up to which it transmits data. Given a certain connectivity requirement, the optimization task is to find a configuration of ranges (or range assignment) of minimal total power consumption providing the required network property. A problem of this kind is called *range assignment problem*.

Three important problems of this type are examined in this thesis. First, we choose a quite abstract approach, allowing arbitrary distance functions without geometrical interpretation. We give the first thorough structural analysis of these problems in different setups. Our results identify easy cases as well as hard ones in terms of complexity as well as various levels of approximability for the individual problems. They also reveal interesting differences between the three problems themselves.

We then turn to geometrical instances, on which there already exists a line of research regarding complexity and approximability in the literature. We contribute to this research by designing new reductions which are more simple and versatile than the ones used before, and produce new and better results. Using our reductions we can solve open problems posed in prior work.

In the last chapter, we turn to approximation algorithms. We give a tight analysis of a well-known approximation algorithm for two of the problems as a function of the input size. A thorough analysis of two natural greedy paradigms is given, with tight results in the general and many special cases. We conclude with the design and analysis of a new approximation algorithm for one problem, and identify the first approximation scheme for some special geometric instances.

Contents

Chapter 1

Introduction

1.1 Background

Imagine the following scenario: You are the owner of an internet café, and you got customers coming and going, everyone bringing a laptop equipped with a wireless local area network (WLAN) card. They come in, sit down at some place where they like it best, take out their laptops and plug them into a power outlet. The WLAN card connects to your server, and the customer is free to surf the internet. After one busy day, while cleaning your coffee machine (an impressive model; your reputation for pouring the best coffee in town attracts an ever-growing number of customers to your place) you begin to wonder how all this network stuff actually works. You have already noticed that all those nice little laptops do after all consume quite some energy. Being very much concerned about saving the environment (and just a little bit about your energy bill), you let your thoughts drift. Do all the laptops have to be connected directly to the server? Would it not be sufficient if some laptops are connected only to other laptops, which are again connected to other laptops etc. which are finally connected to the server? Communication could take place between each laptop and the server using other laptops as intermediate stations. In this way, all laptops would be connected to the server via some other laptops, using much less energy in total. You like your idea and contemplate it further. There seems to be more than one possibility to have all computers connected via such a wireless network. Is it clear what the ideal network consuming the least possible energy in total looks like? When you think about this, you encounter a problem: Maybe you could work out an ideal network for a fixed setting of customer positions. But customers come and go, sitting down at places of their own choice. You practically do not have two identical setups in one day. There is even another problem: Customers have different types of laptops, with different types of WLAN cards. Even if you do have the same setup of people's positions, does it not make a difference at what positions you have someone sitting with an older computer and a dated network

card and where there is someone with a modern computer with a more efficient WLAN card? You abandon these thoughts, as you notice that your coffee machine is at least as clean as when you got it delivered, it is already getting dark outside, and you are beginning to feel hungry. You conclude that it seems plausible that network communication among the laptops and the server should save you (and the world) some energy. But finding the network requiring least total power may not be such an easy task after all...

1.2 Model

In a range assignment problem, we are given a set of sender/transmitter stations and want to establish a wireless network among them with a certain connectivity requirement. Our objective is to minimize the total network power consumption while still providing the required network property. In a widely used model, the power necessary for a sender s to transmit data directly to any other station t at distance at most r is proportional to r^α, where $\alpha > 0$ is called the *distance-power gradient*. In this case we say that t lies within the range of s. In an ideal environment, we would have $\alpha = 2$, but according to [PL95], α may vary from 1 to more than 6, depending on the surrounding environment. The total power consumption of the network is the sum over the energy consumption in all stations.

Another issue is what kind of direct connections we want to establish. A straightforward notion is that we say that a link is established from s to t iff t lies within the range of s. However, some network protocols such as TCP require an immediate acknowledgement from station t to s after each transmitted data packet. Especially in this case, we may want to make sure we only use bidirectional links for data transport. This means that we consider two stations s and t as linked iff t lies within the range of s and vice versa. Both concepts will be formally defined in the next section.

Moreover, even if we had exact (2- or maybe even 3-dimensional) coordinates of our network stations (which is probably not a realistic assumption), it is still not clear in practice how much energy is actually needed to transmit data to another station at a certain distance. There may be such problems as obstacles in the way or interference with other data transmissions. We disregard all these possible problems in the above model, and regard the stations as sitting in a homogenous space, whose properties are modeled completely by the distance-power gradient α.

Having said so, we will not only consider networks with the above setup, which we will later call the *geometric* setting. We will also consider fairly general instances, where the distances between two stations can be quite arbitrary.

At this point, we would like to make a point on the term 'distance'. In the

above, we assumed distances to be the actual Euclidean distances, and the power consumption in a station proportional to the maximal range it transmits across, taken to the power of a constant α. When we define general range assignment problems, we actually want that the distance between two stations is identical to the power required for transmitting between the two stations. Maybe you can compare this to assigning a road between two cities not the actual distance in kilometers, but rather the time needed for travel between the two cities, which may also depend on the type of road etc. The point is that your distances should be measured in the same unit as the resource that you want to optimize.

So in the above geometric setting, network distances are equal to Euclidean distances to the power of α. But we will also consider far more general kinds of distance functions. However, we always do make the following two assumptions:

- All distances are non-negative.
- All distances are symmetric.

The first point seems to be very agreeable, as we probably will not produce energy by transmitting data.The second point, however, does make some assumptions. It implies that at both endpoints of a data link we need the same amount of energy to transmit across this distance. So we do neglect the worries of the concerned internet café owner about a heterogenous set of stations. What we can still model is, e.g., obstacles which may highly increase the power costs between stations whose coordinates lie actually quite close to each other.

Speaking of our example scenario with the café owner, it may be not very realistic nor spectacular. We think, however, that it essentially brings to the point what quantity we want to optimize, namely the total power consumption, and not battery or network lifetime or other resources. Also, we consider the position of stations as given and not adjustable. There exists an overwhelming and fast growing multitude of literature about other optimization problems motivated by wireless networks, which we cannot capture here. For other introductions to this subject, refer to the literature in the bibliography. Scenarios there include archeology, cars, ships and airplanes, satellites and space stations, earthquakes and floods, battlefields and the moon.

1.3 Basic notations and facts

It is now time to more formally define what we mean mathematically when we speak about range assignment problems. We assume basic knowledge of graph theory and computational complexity, as you may acquire from reading, e.g., [KV02] or any other book on these subjects. We will also use some concepts of approximation algorithms, which we try to briefly explain in the appendix, where you can also find references to more extensive introductions to this subject.

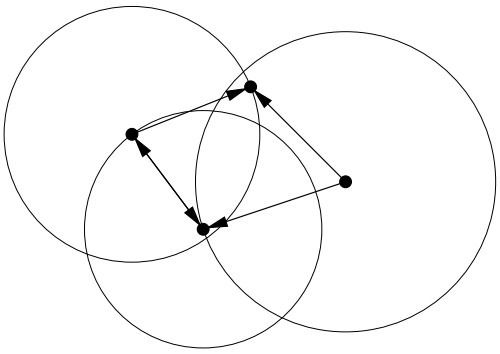

Figure 1.1: A range assignment together with the associated directed communication graph $\vec{G}_r$ ...

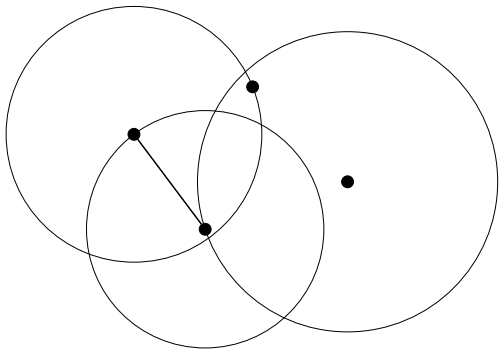

Figure 1.2: ... and the same range assignment with its undirected communcation graph G_r.

Definition 1.1. Let $G = (V, E, d)$ be a weighted graph. A *range assignment on* G is a function $r : V \to \mathbb{R}_+$. A range assignment r defines two subgraphs of G, a directed graph $\vec{G}_r$ and an undirected graph G_r.

- Let $\vec{G}_r = (V, A_r)$ with $A_r = \{(u, v) \mid r(u) \geq d(u, v)\}$. $\vec{G}_r$ is called the *directed communication graph of* r.
- Let $G_r = (V, E_r)$ with $E_r = \{(u, v) \mid \min\{r(u), r(v)\} \geq d(u, v)\}$. G_r is called the *undirected communication graph of* r.

In other words, the undirected communication graph contains exactly those edges whose corresponding arcs appear in A_r in both directions. See Figures 1.1 and 1.2 for examples of both communication graphs for a given range assignment.

The definition of the directed communication graph is quite intuitive. An arc (u, v) is implied by r iff station u can transmit to station v, i.e., v lies in the range of u. A bidirectional (undirected) link in the undirected communication graph is established iff both stations lie inside each other's range. This is a useful scenario for transmission protocols where each transmitted unit of information between two stations is acknowledged by a receipt. Additionally, this ensures that a transmission from node u to node v takes just as long as a transmisson from v to u (at least in theory). This kind of symmetry could be favorable in real-world settings.

With these definitions, we can give a first formulation for range assignment problems.

Definition 1.2 (Range Assignment problems (flavor A)).

Instance: A weighted graph $G = (V, E, d)$ with distances $d : E \to \mathbb{R}_+$, and a certain network property Π.

Task: Assign each node v a radius $r(v)$ such that the communication graph G_r (or $\vec{G}_r$) satisfies property Π, and r minimizes the cost function

$$c(r) = \sum_{v \in V} r(v).$$

Problem definition **(A)** is an intuitive and direct translation from an abstract network problem into the graph world. It is not difficult to see, however, that the above problems can be equivalently reformulated. This is because one can not only go in the one direction from a range assignment r to associated graphs $G_r/\vec{G}_r$, but also in the other direction. I.e., for each subset of arcs/edges $F \subseteq E$, there is a natural minimal range assignment inducing all of F (and possibly more edges).

Definition 1.3 (Range Assignment problems (flavor B)).

Instance: A weighted graph $G = (V, E, d)$ with distances $d : E \to \mathbb{R}_+$, and a certain network property Π.

Task: Find a network $F \subseteq E$ satisfying network property Π that minimizes the cost function

$$c(F) = \sum_{v \in V} \max_{(v,w) \in F} d(v, w)$$

We prefer this formulation as it is more closely related to other, more common network design problems, in which we have to find a "cheapest" subgraph still having a certain network property.

It should be clear how to "go back" from a network F to a range assignment r_F: Define

$$r_F(v) = \max_{(v,w) \in F} d(v, w).$$

It is clear that r_F is the cheapest range assignment inducing F. It is obvious that it does not make sense to assign some node v a range $r(v)$ where there is no node u for which $r(v) = d(v, u)$: Then taking

$$r(v) = \max_{(v,u) \in E} \{d(v, u) \mid d(v, u) < r(v)\}$$

still induces the same communication graph, at lower cost.

The equivalence of formulations **A** and **B** is elementary and has a nice and compact formalization: Let F_r indicate the arc resp. edge set of communication graph G_r resp. $\vec{G}_r$. With this, we have

$$\begin{array}{rcll} r_{F_r}(v) & \leq & r(v) & \forall\ v \in V, \\ F_{r_F} & \supseteq & F & \forall\ F \subseteq E. \end{array}$$

So, the operations G_r and r_F make the search for a cheapest range assignment r and network F completely exchangable, which we shortly state here:

Observation 1.4. *Flavor* **A** *and* **B** *of the* Range Assignment Problem *are equivalent.*

The author's intuition of this problem tastes more like flavor **B**, while both interpretations will be used frequently.

Let us have a closer look at the cost function $c : 2^E \to \mathbb{R}_+$, defined as

$$c(F) = \sum_{v \in V} \max_{(v,w) \in F} d(v, w). \tag{1.1}$$

c goes through all nodes and counts only the longest incident edge to (resp. outgoing arc from) each node. Compare this to a more common cost function, the sum of weights of its elements. We denote this cost function by $|\cdot|$, meaning $|F| = \sum_{e \in F} d(e)$. In accordance with this notation, we sometimes use $|e| = d(e)$ synonymously.

Now note that for an undirected edge set $F \subseteq E$, $|F|$ can also be written as

$$|F| \;=\; \frac{1}{2} \sum_{v \in V} \sum_{\{v,u\} \in F} d(v, u), \tag{1.2}$$

and for a directed set of arcs $F \subseteq A$ as

$$|F| \;=\; \sum_{v \in V} \sum_{(v,u) \in F} d(v, u). \tag{1.3}$$

When we compare the latter two expressions for $|\cdot|$ with the one for $c(\cdot)$ in Equation 1.1, the only difference (apart from a constant factor $\frac{1}{2}$) is that we have taken the max instead of the $\sum$ over the same sets. One could interpret this as a transition from a sum-norm to a maximum-norm. As a fact, instead of a linear cost function, we now have a non-linear one which very often makes optimization much more difficult. Via this change of cost function, we can define a range assignment version of almost any network design task.

We now introduce the concrete range assignment problems this thesis is about. In short, we investigate the generic Range Assignment Problem with property Π being connected, strongly connected and rooted arborescence. Let us state these three problems formally.

Definition 1.5 (Connectivity**).**

Instance: A weighted graph $G = (V, E, d)$ with distances $d : E \to \mathbb{R}_+$.

Task: Find a spanning tree $F \subseteq E$ which minimizes

$$c(F) = \sum_{v \in V} \max_{\{v,w\} \in F} d(v, w).$$

Definition 1.6 (Strong Connectivity).

Instance: A weighted graph $G = (V, E, d)$ with distances $d : E \to \mathbb{R}_+$.

Task: Find a strongly connected (directed) network $F \subseteq E$ which minimizes

$$c(F) = \sum_{v \in V} \max_{(v,w) \in F} d(v, w).$$

Definition 1.7 (Broadcast).

Instance: A weighted graph $G = (V, E, d)$ with distances $d : E \to \mathbb{R}_+$, and a specified node $s \in V$.

Task: Find a spanning arborescence $F \subseteq E$ rooted in s which minimizes

$$c(F) = \sum_{v \in V} \max_{(v,w) \in F} d(v, w).$$

Note that by allowing only undirected graphs G as instances, we implicitly assume symmetric distances. The solutions for Connectivity are undirected networks, the solutions for Strong Connectivity and Broadcast are directed networks.

Observe further that the three problems are given in "decreasing strength", as each problem is a relaxation of the one before. I.e., for all instances G, every feasible range assignment r for Connectivity is feasible for Strong Connectivity, and every feasible range assignment for Strong Connectivity is again feasible for Broadcast (for every possible source node $s \in V$).

Observation 1.8. *Let $opt(G, \Pi)$ denote the value of an optimal range assignment on G for network requirement Π. For any range assignment instance G, we have*

$$opt(G, \text{Broadcast}) \leq opt(G, \text{Strong Con.}) \leq opt(G, \text{Connectivity}).$$

Let us consider the above problems with the usual cost function $|\cdot|$. Recall the following known complexity results, which can be found in many textbooks on combinatorial optimization, see, e.g., [KV02]. Connectivity with $|\cdot|$ is known as the Minimum Spanning Tree problem, and very efficient exact algorithms are known (at least) since the 1950s, namely Prim's algorithm [Pri57] and Kruskal's algorithm.[Kru56] Broadcast with $|\cdot|$ is known as the Minimum Weight Rooted Arborescence Problem and is equivalent to the Maximum Weight Branching Problem. It can be solved in polynomial time using Edmonds' Branching Algorithm [Edm67]. On the contrary, Strong Connectivity with $|\cdot|$, known as the Minimum Strongly Connected Spanning Subgraph Problem, is NP-hard already on unweighted graphs: Indeed,

a graph G has a strongly connected spanning network with n edges iff it contains a Hamiltonian cycle.

Under the cost function $c(\cdot)$, all three problems become NP-hard in general as well as in some special cases. The next two chapters will deal with those results. We close this section with some important basic properties of these problems.

Lemma 1.9. *Let $G = (V, E, d)$ be an instance of* (Strong) Connectivity. *For a feasible range assignment r,*

$$r(v) \geq \min_{(u,v)\in E} d(u, v)$$

must hold for all $v \in V$.

Proof. If there was some $v \in V$ with $r(v) < \min_{(u,v)\in E} d(u, v)$, v could have no incident edge (resp. outgoing arc) in G_r (resp. $\vec{G}_r$), implying that r is infeasible. □

As simple as this Lemma may be, we will need it frequently later in our proofs. It directly motivates the following

Definition 1.10. Let $G = (V, E, d)$ be an instance of (Strong) Connectivity. For each $v \in V$, set

$$r_{min}(v) = \min_{(u,v)\in E} d(u, v).$$

r_{min} is called the *minimal range assignment* (or *minimal configuration*) of G.

After this lower bound for the radius of each single node, we provide a fundamental lower bound on the value of an optimal solution for (Strong) Connectivity.

Lemma 1.11. *Let $G = (V, E, d)$ be an instance of* (Strong) Connectivity. *Let MST be a minimum spanning tree of G, and $mst = |MST|$ its length. Let $opt(G)$ be the cost of an optimal range assignment r^* for G. We have:*

$$opt(G) \;\geq\; mst + \max_{e\in MST} d(e).$$

Proof. It suffices to prove the lower bound for Strong Connectivity; it will also hold for Connectivity due to Observation 1.8.

Let $t \in V$ be an arbitrary fixed node. As $\vec{G}_{r^*}$ is strongly connected, it must contain a network of paths from *all* other nodes towards t. Let T_t be such a network (it need not be unique). In this network, every node v has exactly one outgoing arc (v, u), where u is the next node on the v-t path in T_t (except for t, which has no outgoing arc in T_t). See Figure 1.3 for an illustration.

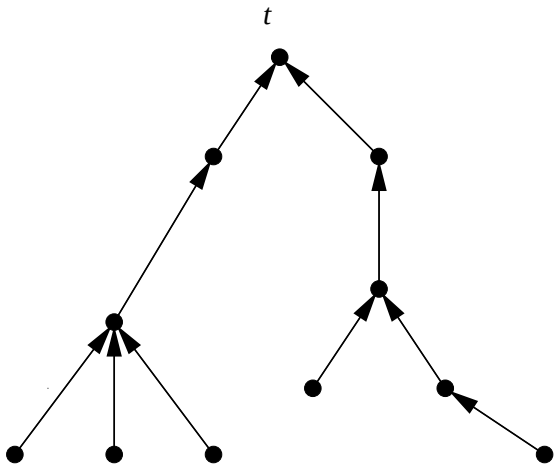

Figure 1.3: T_t, a subnetwork of paths to t. Each radius must dominate the outgoing arc.

Let $e(v) = (v, u) \in T_t$ for $v \neq t$. T_t witnesses that $r^*(v) \geq |e(v)|$ for each $v \in V \setminus \{t\}$. Summing up yields

$$c(\vec{G}_{r^*}) \geq c(T_t) + r(t) \geq |T_t| + r(t).$$

Now notice that when we regard the arcs of T_t as (undirected) edges, T_t is a spanning tree of G, thus $|T_t| \geq mst$. This holds for *any* $t \in V$, so we can choose t such that $r^*(t) = \max_{v \in V} r^*(v)$. This corresponds directly to the longest arc $(t, s) \in \vec{G}_{r^*}$, so we have $r(t) = d(t, s)$. Again, as $\vec{G}_{r^*}$ needs to contain some spanning tree, say T, $d(t, s) \geq \max_{e \in T} d(e)$ must hold. As the spanning trees of a graph form a matroid, we know that the longest edge in any spanning tree T is at least as long as the longest edge in an MST. (Else, the base exchange property would allow us to swap the longest edge in MST for some cheaper edge in T, making MST even cheaper, a contradiction.) So we have

$$r(t) = d(t, s) \geq \max_{e \in T} d(e) \geq \max_{e \in MST} d(e),$$

proving our claim. □

We do not have such a nice lower bound for Broadcast. E.g., a star with $n - 1$ nodes around source s, all edges of weight 1, has a feasible Broadcast range assignment of constant cost 1. This already partly explains the lack of good (constant factor) approximations for Broadcast, as good lower bounds are a key ingredient for approximation algorithms. In the next chapter, we will see that Broadcast is indeed *provably* harder to approximate than the other two range assignment problems.

1.4 Outline

We roughly sketch the outline of this thesis. More detailed summaries and discussions can be found at the end of each chapter.

In chapter 2, we analyze the computational complexity of range assignment problems in very abstract settings. We examine how much (or how little) and what kind of structure the distance function may have such that the addressed range assignment problems remain efficiently solvable, and at what point they become hard to solve or approximate. This is the first thorough treatment of this kind for these problems. This may be due to the fact that the study of wireless network design problems is relatively young and very much application driven. Non-geometrical instances have, to our knowledge, not yet been widely considered. More classical network design and routing problems have been treated with similar studies, e.g. the Traveling Salesman Problem [PY93] and the Steiner Tree Problem [BP89]. Our results help in understanding what essentially makes range assignment problems hard, and they reveal structural differences between different kinds of range assignment problems that may seem very similar at first.

Chapter 3 is about computational and approximation hardness of geometric range assignment problems. There already exists a line of research on this subject, which we continue by answering some open questions posed in prior work. Our main contribution are new reductions which are technical actually less involved and more elegant than earlier constructions, besides leading to new and/or improved results. Their relative simplicity also brings about higher flexibility, so they may be adaptable for other problems in this area.

In Chapter 4, we consider approximation algorithms for these problems. We start with a tight analysis of the now well-known MST-heuristic as a function of the input size. Then we give a quite thorough treatment of two natural greedy heuristics for the Connectivity problem, giving tight analyses in general as well as for many important special settings. We conclude with an examination of Strong Connectivity, for which we identify the first PTAS in specific geometric settings. We conclude with a new approximation algorithm for this problem.

Chapter 2

Complexity results

2.1 Results for (Strong) Connectivity

We now give various results on the hardness of different flavors of the Connectivity and Strong Connectivity problems. Quite often, results for the Connectivity and Strong Connectivity problems are very similar, as are their proofs. Reductions for the Connectivity problem frequently work identically for the Strong Connectivity problem. Or at least they yield a solid skeletal structure which, by adding some additional (and often rather technical) details, can also be used for the Strong Connectivity problem.

All reductions are from the Set Cover problem or the Vertex Cover problem, an important special case of the Set Cover problem. The Set Cover problem is defined as follows.

Definition 2.1 (Set Cover Problem).

Instance: A set $S = \{1, \ldots, n\}$ of n elements, and a collection $\mathcal{S} = \{S_1, \ldots, S_m\}$ of m subsets of S.

Task: Cover the set S using as few sets S_i as possible, i.e., find a smallest subset of $\mathcal{S}$ whose union is S.

Set Cover is among Karp's original NP-complete problems.[Kar72] It is not approximable within $c \log n$ for some constant c, unless P = NP.[RS97] Feige showed in [Fei98] that it is not approximable within $(1 - \varepsilon) \log m$ for any $\varepsilon > 0$, unless NP has $n^{O(\log \log n)}$-time deterministic algorithms.

Theorem 2.2. (Strong) Connectivity *is NP-hard.*

Proof. We use a rather generic reduction from the Set Cover problem. Given an instance of Set Cover with elements $\{1, \ldots, n\}$ and subsets $\{S_1, \ldots, S_m\}$, we construct a graph as follows. Let $V = \{u, w_1, \ldots, w_m, v_1, \ldots, v_n\}$ be the set of

vertices. As the notation may suggest, node v_i corresponds to element i and node w_j to subset S_j. Accordingly, we call the v_is *element nodes* and the w_js *set nodes*. The set of edges E contains all edges $\{u, w_j\}$ for $j \in \{1, \ldots, m\}$. Furthermore, an edge $\{w_j, v_i\}$ is contained in E iff $i \in S_j$. For an edge $e = \{u, w_j\}$, we define $d(e) = 1$. For edges $e = \{w_j, v_i\}$, we set $d(e) = 2$. See Figure 2.1 for an illustration of this reduction.

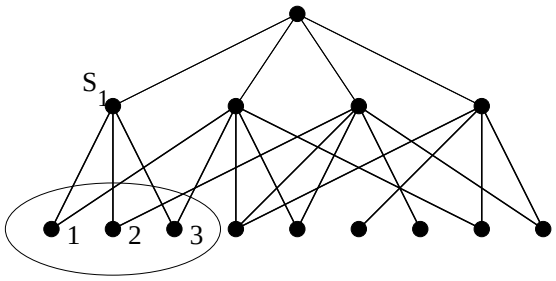

Figure 2.1: An illustration of a reduced Set Cover instance, with $S_1 = \{1, 2, 3\}$, etc.

In the minimal configuration r_{min}, we have that $r_{min}(u) = r_{min}(w_j) = 1$ for all j, and $r_{min}(v_i) = 2$ for all i.

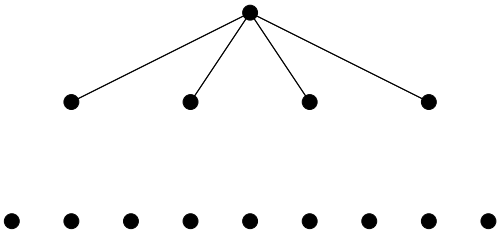

Figure 2.2: The undirected communication graph $G_{r_{min}}$...

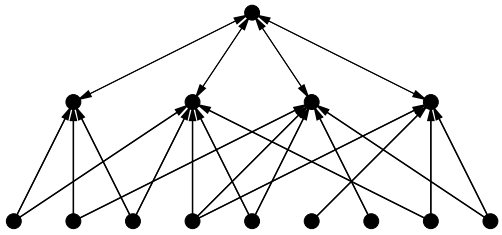

Figure 2.3: ... and the directed communication graph $\vec{G}_{r_{min}}$.

The only way to add more edges to the minimal configuration is to increase $r_{min}(w_j)$, for some of the js, from 1 to 2. (Increasing some radius above 2 would only add to the total costs, but no new edges.) Let r be a feasible range assignment, and $C_r = \{j \mid r(w_j) = 2\}$. An edge from a w_j with $j \in C_r$ to v_i is obviously established iff $i \in S_j$. Note that as the communication graph of r is (strongly) connected, C_r has to constitute a set cover in the original instance. Thus, the total power consumption of r amounts to

$$\sum_{v_i \in V} r(v_i) = m + 1 + 2n + |C_r| \tag{2.1}$$

If we were given an optimal range assignment r^* for the reduced instance, we could easily determine the associated set C_{r^*}. C_{r^*} has to be a set cover of minimal

cardinality; otherwise, r^* would not be optimal. This means that an efficient algorithm for (STRONG) CONNECTIVITY would also imply the existence of an efficient algorithm for SET COVER, and thus P = NP. □

Being NP-hard in general, it is still interesting to know in which settings these problems remain NP-hard, and at which point they become easy. One might also like to see if there is some case where the complexity status of CONNECTIVITY and STRONG CONNECTIVITY differs, drawing a line between them. We shall identify such a case later.

More precisely, we investigate how much structure the distance function d needs to make the problems difficult. As an example, the TRAVELING SALESMAN PROBLEM [PY93] and the STEINER TREE PROBLEM in graphs [BP89] remain NP-hard (in fact, even APX-hard) on complete graphs where the distance between two distinct points is either 1 or 2. STEINER TREE also remains NP-hard in bipartite graphs where all edges have weight 1 [GJ79].

Definition 2.3. We define problem $(d_1, d_2, \ldots, d_k)$-(STRONG) CONNECTIVITY/BROADCAST to be the respective problem restricted to instances where the underlying graph is complete, and we have that $d : V \times V \to \{d_1, d_2, \ldots, d_k\}$.
With $(d_1, d_2, \ldots, d_k, \infty)$-(STRONG) CONNECTIVITY/BROADCAST, we refer to the respective problem as above where the graph need not be complete.

So in our new notation, we have stated above that the $(1, 2)$-Steiner tree problem and the $(1, \infty)$-Steiner tree problem in bipartite graphs are NP-hard.

Corollary 2.4. *The proof of Theorem 2.2 shows that already the* $(1, 2, \infty)$-(STRONG) CONNECTIVITY *problem is NP-hard.*

Observation 2.5. *The* (1)- *and the* $(1, \infty)$-(STRONG) CONNECTIVITY *problem is trivial.*

Obviously, each radius must be at least 1. On the other hand, all existent edges are already induced by this range assignment. So if it not feasible (which may be the case in the $(1, \infty)$ version), this is because the input graph itself is not connected, so no range assignment will be feasible at all.

Let us see what happens when we allow a second weight. At least, (STRONG) CONNECTIVITY will not be completely trivial anymore. However, it is still quite easy.

Theorem 2.6. *The* $(1, 2)$-(STRONG) CONNECTIVITY *problem is in P.*

Proof. We give a simple efficient algorithm for this problem. Start with the range assignment which assigns radius 1 to each vertex. This assignment will induce some (strongly) connected components. (For the directed case, note that there is an arc (v, w) iff there is an arc (w, v), so the strongly connected components are

identical to the connected components.) If this range assignment leaves us with one component, we are done already.

Otherwise, in each component C_i, we need at least one vertex with radius 2, because we have identified a cut $(C_i, V \setminus C_i)$ on which all edges have distance 2. But if we simply choose one arbitrary vertex v_i per component C_i to have radius 2, we will get a complete graph on the vertices v_i, cf. Figure 2.4.

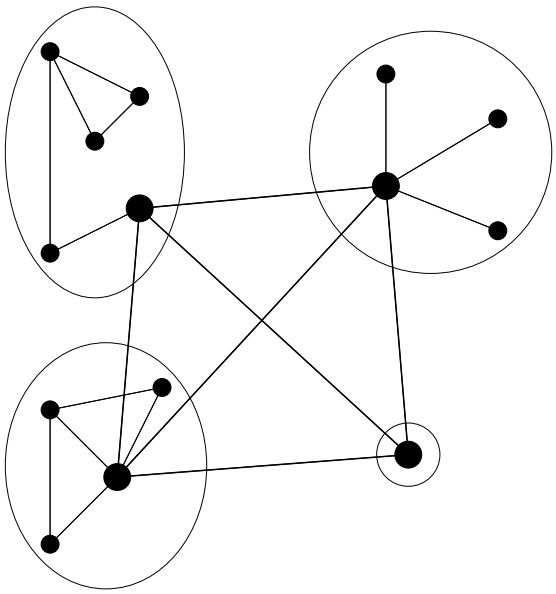

Figure 2.4: The radius 1 components, connected via radius 2 nodes

In this way, we have interconnected the components C_i at minimum possible cost. □

We now investigate the complexity status of the $(1, 2, 3)$-(Strong) Connectivity problem. One might suspect that the step from 2 to 3 (different distances, in this case) may be the step from a very simple problem to a NP-hard one, a common phenomenon in combinatorial optimization. This suspicion is nourished by the fact that both problems are hard in the $(1, 2, \infty)$ setting, which is also some sort of a three-weight szenario. One might also think that the proofs should be very similar.

It is indeed true that both problems are NP-hard in the $(1, 2, 3)$-setting. Also, the construction for $(1, 2, 3)$-Connectivity is nearly identical to the one for for $(1, 2, \infty)$-Connectivity; we only need to argue a little more carefully about its correctness.

What might be more interesting is that our Set Cover construction does not work for $(1, 2, 3)$-Strong Connectivity. Instead, we will use a reduction from the Hamiltonian Cycle problem. Again, this will work for this problem

only, not its directed counterpart. Another curiosity is that for $(1,2,3)$-Strong Connectivity, the optimal solution can only take one of two values, a certain (easily computable) k, or $k+1$. So, informally speaking, one might say $(1,2,3)$-Strong Connectivity is "NP-hard but not so hard". Later in this section, we can make this statement more precise by means of approximation preserving reductions.

Theorem 2.7. *The* $(1,2,3)$-Connectivity *problem is NP-hard.*

Proof. As above, for a Set Cover instance with elements $\{1,\dots,n\}$ and subsets $\{S_1,\dots,S_m\}$, we use the node set $V = \{u, w_1,\dots,w_m, v_1,\dots,v_n\}$, and we have $E = V \times V$. We define distances as follows:

$$d(e) = \begin{cases} 1 & \text{for } e = \{u, w_j\}, \quad j = 1,\dots,m, \\ 2 & \text{for } e = \{w_j, v_i\}, \quad i \in S_j, \\ 3 & \text{else.} \end{cases}$$

Note that this is nearly the same reduction as in Theorem 2.2. The only difference is that we now have a complete graph, and where there were no edges in the generic case, we have edges of weight 3.

To prove that this reduction works correctly, we only need to show that in an optimal range assignment, no vertex can have a radius larger than 2. Once we know this, we can follow the argumentation from the proof of Theorem 2.2.

Lemma 2.8. *In an optimal range assignment for the above construction, no vertex can have a radius larger than 2.*

Assume on the contrary that there is an optimal range assignment where some vertices have radius 3. (Having a larger radius than 3 would not induce any additional edges, but only be more expensive.) Let $\varnothing \neq X \subseteq V$ be the set of those vertices. We now construct a cheaper range assignment which remains connected. The construction works as follows. For each $v_i \in X$, decrease its radius to 2. If this disconnects v_i from the rest of the network, increase some w_j with $d(v_i, w_j) = 2$ (i.e., $i \in S_j$,) to 2. There must be some such w_j, or else the original Set Cover instance is infeasible (and this condition is easy to verify). Note that this procedure can only make the total range assignment cheaper. Afterwards, for each $w_j \in X$, we simply decrease its radius to 2. As now all v_i have radius 2, and the w_j and u are connected in a minimal configuration, this does not delete any edges from the network. Similarly, if $u \in X$, we can set its radius to 1.

Finally, we notice that if all vertices in X are element vertices, at least one of them, say v_i, there must have been some w_j with $i \in S_j$ which already had radius 2. Otherwise, X could not be connected to the rest of the graph. Thus, the above procedure strictly decreases the total cost of the supposedly optimal

range assignment, showing that no radius can be larger than 2 in an optimal range assignment.

□

The above reduction fails for the STRONG CONNECTIVITY case. When we apply the identical construction, the optimal strongly connected range assignment is trivial: Take the minimal configuration, and increase any one radius of the set nodes w_j or u from 1 to 3, see figure 2.5.

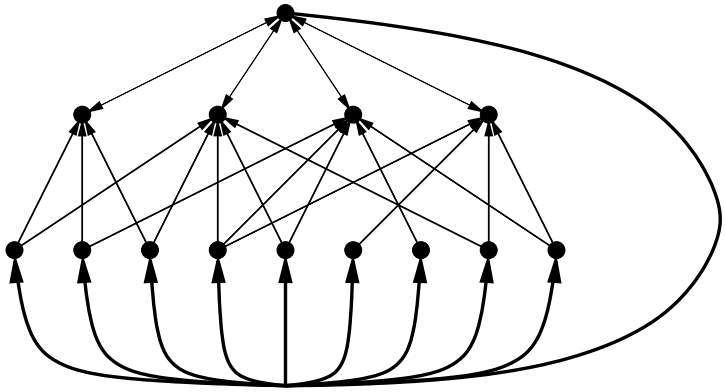

Figure 2.5: Setting $r(u) = 3$ makes a minimal configuration strongly connected. (Not all arcs are displayed here.)

The resulting range assignment is strongly connected. This not possible for any cheaper range assignment. (Except if there is one set $S_j \supseteq V$ covering all elements, but such an instance of SET COVER would be trivial.)

Before we describe the new reduction, let us first formulate a structural property of $(1, 2, 3)$-STRONG CONNECTIVITY which distinguishes from its undirected counterpart.

Lemma 2.9. *An optimal solution of* $(1, 2, 3)$-STRONG CONNECTIVITY *can only take one of two possible values k or $k + 1$, where k is easily calculable. Furthermore, instances with a cut consisting only of cost 3 edges (i.e., an MST contains a cost 3 edge,) can be solved efficiently.*

Proof. Let $G = (V, V \times V, d : V \times V \to \{1, 2, 3\})$ be a $(1, 2, 3)$-STRONG CONNECTIVITY instance. All nodes have radius at least 1. Setting all radii to 1, i.e., range assignment $r = \mathbb{1}$, constitutes some strongly connected components. Let $\mathcal{C}^1$ be the partition of V into strongly connected components in this range assignment. We call the elements of $\mathcal{C}^1$ the '$\mathbb{1}$-components'. It is convenient to regard $\mathcal{C}^1$ as the node set of an auxiliary graph G^1, i.e., we shrink each strongly connected component into one supernode. In $\hat{G}$, we draw an edge $\{C_i^1, C_j^1\}$ iff there exist nodes $v \in C_i^1$, $w \in C_j^1$ with $d(v, w) = 2$. This (undirected) graph again has some

connected components; let $\mathcal{C}^2$ be the partition of $\mathcal{C}^1$ into these components. Let F be a spanning forest of $\hat{G}$, i.e., a collection of spanning trees in each component in $\mathcal{C}^2$. Arbitrarily choose one (auxiliary) node in each component, and direct the edges of the corresponding spanning tree towards it. I.e., in each component $C_i^2 \in \mathcal{C}^2$ we have a node $\hat{v}_i \in \mathcal{C}^1$ such that there is a directed path towards $\hat{v}_i$ from any other node in component C_i^2. See Figure 2.6 for an illustration of the constructed graph we call $\hat{G}'$.

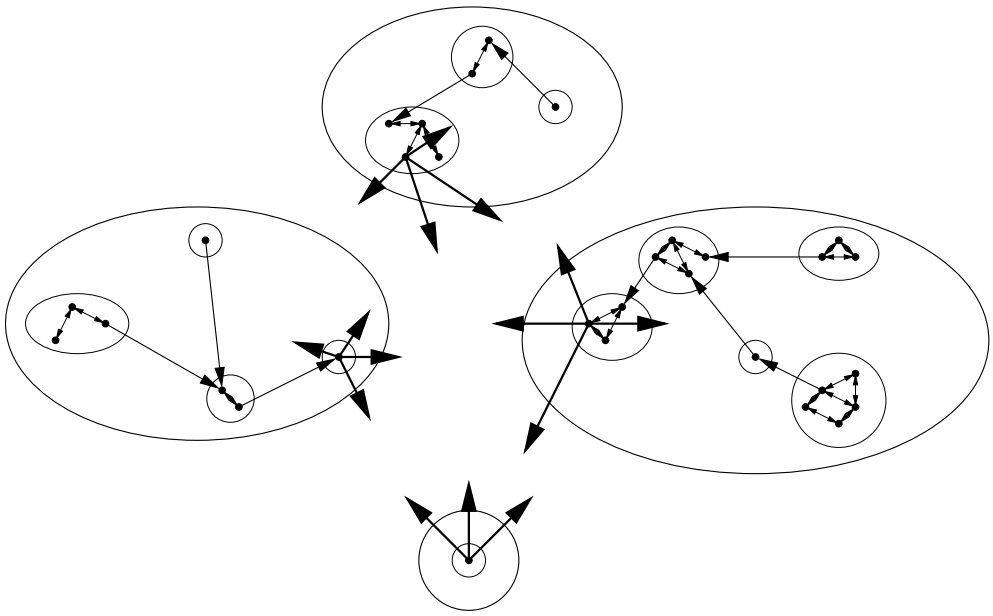

Figure 2.6: Graph $\hat{G}'$ with an optimal range assignment.

Using $\hat{G}'$, we construct a range assignment in the following way. For each arc (C_i^1, C_j^1) in $\hat{G}'$, set the radius of the corresponding tail node v_i^2 (i.e., we have $v_i^2 \in C_i^1$ with $d(v, w) = 2$ for some $w \in C_j^1$) to 2. Additionally, we arbitrarily choose one node $v_j^3 \in \hat{v}_j$ for the selected supernode $\hat{v}_j$ from each component $C_j^2 \in \mathcal{C}^2$, and set its radius to 3. We claim that the resulting range assignment is strongly connected, and (nearly) optimal.

To see that we have strong connectivity, consider any node $v \in V$ of G. Surely, there is a directed path to all nodes in its component $C_i^1 \ni v$. In particular, there is such a path to v_i^2. By construction, there is a directed path from v_i^2 into the supernode $\hat{v}_j$ in the same $\hat{G}$ component, i.e., $v_i^2 \in C_k^2$ and $\hat{v}_j \in C_k^2$, for some k. So in total, we have a directed path from v to "its" v_j^3. But this node has radius 3, so it has an arc to every other node in G, meaning there is a directed path from v to all other nodes, for an arbitrary $v \in V$.

To show optimality, notice that in each $\mathcal{C}^1$-component, we need at least one node of radius at least 2, and in every $\mathcal{C}^2$, we need at least one node of radius 3.

So we achieve strong connectivity at least possible cost.

Now notice that the above only holds in case there is more than one component in $\mathcal{C}^2$. If we have only one component in $\mathcal{C}^2$, it is not immediate whether we actually need one node of radius 3 to reach other components. But we do need one radius 2 node per component/supernode in $\mathcal{C}^1$, so a strongly connected range assignment has to cost at least as much as our constructed assignment minus 1 unit.

□

So, in case the largest edge of an MST has distance 2 (distance 1 being trivial), it is unclear whether one radius 2 node per component in $\mathcal{C}^1$ might suffice in the above construction if we choose them in a clever way. We now show that this is actually an NP-hard choice.

Theorem 2.10. *The* $(1, 2, 3)$-Strong Connectivity *problem is NP-hard.*

Proof. We use a reduction from the Hamiltonian Cycle problem, i.e., the problem to decide whether a graph $G = (V, E)$ contains a cycle through all the vertices in V. This problem is one of Karp's classical NP-complete problems.[Kar72] Given a Hamiltonian Cycle instance $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$, we construct a $(1, 2, 3)$-Strong Connectivity $\hat{G} = (\hat{V}, \hat{V} \times \hat{V}, d)$ instance as follows. For each edge $\{v_i, v_j\} \in E$, we create a node $\hat{v}_{ij}$ and a node $\hat{v}_{ji}$. These are all nodes in $\hat{V}$. Distances are defined as follows:

$$d(\hat{v}_{ij}, \hat{v}_{\tilde{i}\tilde{j}}) = \begin{cases} 1 & \text{if } i = \tilde{i}, \\ 2 & \text{if } i = \tilde{j} \text{ and } j = \tilde{i}, \\ 3 & \text{else.} \end{cases}$$

In other words, we have replaced each node v_i of the Hamiltonian Cycle instance by a "supernode" in the sense of the above proof, containing one endpoint for each incident edge in E. The vertices in each supernode have distance one to another, while the distance 2 edges in $\hat{G}$ are the original edges from E. See Figure 2.1 for an example.

A strongly connected range assignment has to have at least radius 1 on each node, and one radius 2 node per component/supernode in $\mathcal{C}^1$, as constructed in the above proof. Note that these supernodes in $\hat{G}$ correspond to the nodes in G by construction. We have two nodes in $\hat{V}$ per original edge in E, and one component per original node in V, thus a feasible range assignment $\hat{r}$ for $\hat{G}$ has to cost at least

$$\sum_{v \in V} \hat{r}(v) \quad \geq \quad 2|E| + |V| =: k$$

If we apply the above algorithm on $\hat{G}$, i.e., we choose any vertex $\hat{v} \in V$ to have radius 3 and direct an MST on the supernodes towards the supernode of $\hat{v}$, we get a strongly connected range assignment costing $k + 1$.

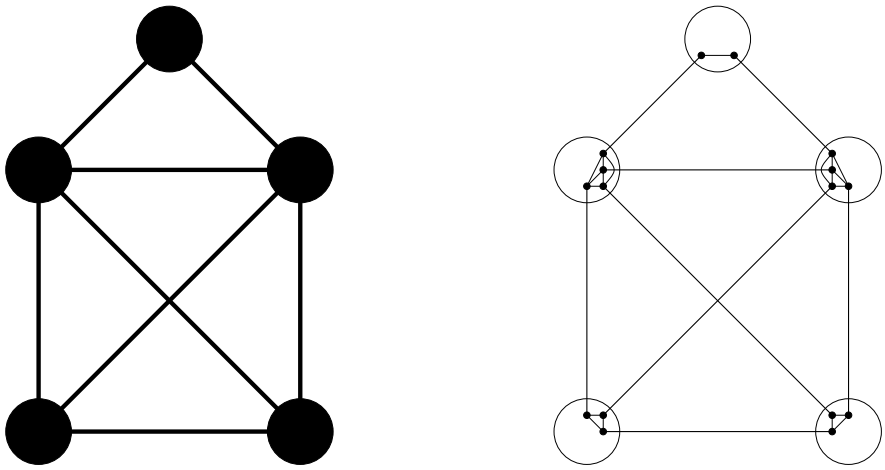

Figure 2.7: An instance G of HAMILTONIAN CIRCUIT on the left, and the resulting $(1,2,3)$-STRONG CONNECTIVITY instance $\hat{G}$ on the right. Thin edges have distance 1, bold edges distance 2. Distance 3 edges are not displayed. The circles on the right mark the 'supernodes' as constructed in the above proof.

We now show that there exists a Hamiltonian Circuit in G iff there is a feasible range assignment of cost k. One implication is easy: Suppose we have a Hamiltonian Cycle H in G, consisting of edges

$$H = \{\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}\}.$$

(We assume this cycle for notational convenience.) Now consider the following range assignment of cost k:

$$\hat{r}(\hat{v}_{ij}) = \begin{cases} 2 & \text{for } j \equiv i+1 \mod n, \\ 1 & \text{else.} \end{cases}$$

See Fig. 2.8 for an example Hamiltonian Cycle and the corresponding range assignment.

Now, for any pair of nodes $(\hat{v}_{ij}, \hat{v}_{\tilde{i}\tilde{j}})$, in our range assignment we have the directed path (which alternates on weight 1 and 2 arcs):

$$\hat{v}_{i,j} \to \hat{v}_{i,i+1} \to \hat{v}_{i+1,i} \to \hat{v}_{i+1,i+2} \to \hat{v}_{i+2,i+1} \to \ldots \to \hat{v}_{\tilde{i}-1,\tilde{i}} \to \hat{v}_{\tilde{i},\tilde{i}-1} \to \hat{v}_{\tilde{i},\tilde{j}}$$

so we have strong connectivity.

On the other hand, assume $\hat{G}$ has a feasible range assignment $\hat{r}$ of cost k. In other words, we have a strongly connected range assignment with exactly one radius 2 node in each $\mathbb{1}$-component (and all other nodes have radius 1). Note that every node $\hat{v}_{ij} \in \hat{V}$ has exactly one edge of distance 2 adjacent to it, namely $(\hat{v}_{ij}, \hat{v}_{ji})$. This means we have exactly one arc per $\mathbb{1}$-component giving

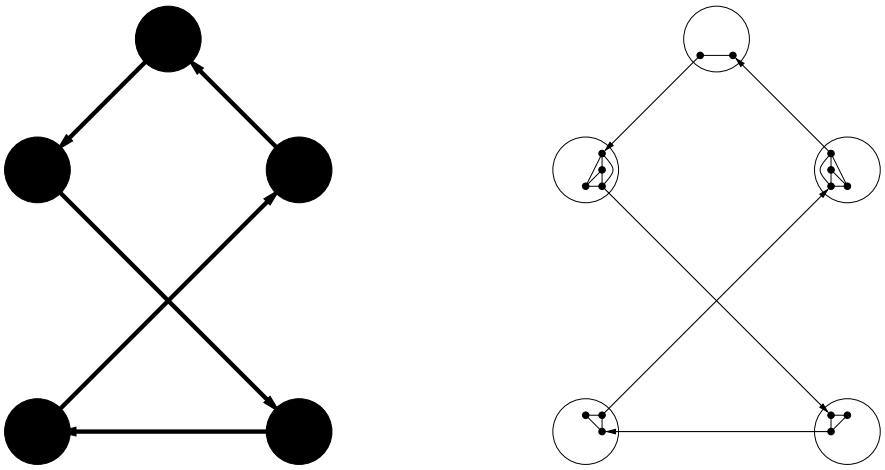

Figure 2.8: A Hamiltonian cycle in the above example graph, and the constructed range assignment.

us strong connectivity. Now note that this is only possible if these $|V|$ arcs form a Hamiltonian Cycle on the supernodes. This is the only way to ensure strong connectivity with n edges on n nodes. I.e., the set

$$H_{\hat{r}} := \{\{v_i, v_j\} \mid \hat{r}(\hat{v}_{ij}) = 2\}$$

has to be a Hamiltonian Cycle in G in this case. □

Lemma 2.9 allows us to formulate an approximation scheme.

Corollary 2.11. *There exists a PTAS for the* $(1, 2, 3)$-Strong Connectivity *problem.*

Proof. We construct a polynomial $(1 + \varepsilon)$-approximation algorithm for any constant $\varepsilon > 0$. The algorithm in Lemma 2.9 always gives us a solution of value $k + 1$, whereas we know an optimal solution has cost at least k. This means our algorithm is a $\frac{k+1}{k}$-approximation algorithm. For $k \geq 1/\varepsilon$ this is already a $(1+\varepsilon)$-approximation, so we assume $k < 1/\varepsilon = O(1)$. We have that $k = |MST| + 2 \geq n$, so in particular $n = O(1)$. For graphs with a constant number of nodes, we can try every sensible range assignment with cost k by brute force and stay polynomial. That is, in every $\mathbb{1}$-component, we simply try out each node to have radius 2 in turn and check whether we obtain a strongly connected graph. Assume we have ℓ $\mathbb{1}$-components of sizes $c_1, c_2, \ldots, c_\ell$, where $c_1 + c_2 + \ldots + c_\ell = n$. The number of possibilities to choose one node in each component simultaniously is

$$c_1 \cdot c_2 \cdot \ldots \cdot c_\ell \leq n^\ell \leq n^n \leq O(1),$$

roughly bounded.

We get the same rough bound just if we try every possibility to choose ℓ nodes to have radius 2, for which there are

$$\binom{n}{\ell} \leq n^{\ell}$$

possibilities. □

We note that this NP-hardness construction for $(1,2,3)$-Strong Connectivity does not work for $(1,2,3)$-Connectivity. The problem is that for connectivity, we need both endpoints of a distance 2 edge to have radius 2, so we would need two such nodes per supernode. But we can connect all supernodes if we just increase *one* node per component to radius 2 or 3.

Remark 2.12. Let $G = (V, E)$ be a connected graph, and $\hat{G}$ the auxiliary graph constructed in the NP-hardness proof for Strong Connectivity. An optimal range assignment $\hat{r}$ for the $(1,2,3)$-Connectivity problem on $\hat{G}$ can be efficiently constructed. It has cost

$$c(\hat{r}) \;=\; 2(|E| + |V|) - \max\{2, |M|\},$$

where M is a maximum matching in G.

Proof. Each node needs radius at least 1. This sums up to a basic power consumption of $2|E|$. Now, we have cliques inside the supernodes, and no other edges.
We distinguish two cases:
(i) $\forall$ nodes $\hat{v}_{ij}$, we have $\hat{r}(\hat{v}_{ij}) \leq 2$.
This means we establish connectivity between the supernodes with distance 2 edges. To connect supernodes v_i and v_j, we need to set $\hat{r}(\hat{v}_{ij}) = \hat{r}(\hat{v}_{ji}) = 2$. So in this case a minimal connected range assignment $\hat{r}$ directly corresponds to some spanning tree of G, with additional cost $2(|V| - 1)$.
(ii) $\exists$ a node $\hat{v}_{ij}$ with $\hat{r}(\hat{v}_{ij}) = 3$.
We first argue that we can assume, wlog., that every supernode has *exactly* one node of radius larger than 1. (Of course, every supernode needs at least one such node.) Let us first deal with a supernode v_i containing a range 3 node $\hat{v}_{ij}$. It is immediate that it cannot contain another range 3 node $\hat{v}_{i\tilde{j}}$, so assume it contains a range 2 node $\hat{v}_{i\tilde{j}}$, inducing a distance 2 edge that connects supernodes v_i and $v_{\tilde{j}}$. In this case, we can set

$$\hat{r}(\hat{v}_{i\tilde{j}}) := 1 \quad \text{and} \quad \hat{r}(\hat{v}_{\tilde{j}i}) := 3,$$

without losing connectivity.

We call supernodes with a range 3 node *3-supernodes*. We now know we can assume that 3-supernodes do not contain any other node of range greater

than 1. Now consider the graph resulting from removing 3-supernodes. This graph falls into components. If all these components were trivial, i.e., consisting of only one supernode, we were done already, so assume this is not the case. Consider one such component and imagine it rooted at the supernode which has a distance 2 edge to a 3-supernode. (There has to be exactly one such supernode per component.) As long as this component is non-trivial, we do the following. Choose one supernode with exactly one range 2 node $\hat{v}_{ij}$ (i.e., a leaf of this tree). Now set

$$\hat{r}(\hat{v}_{ij}) := 3 \quad \text{and} \quad \hat{r}(\hat{v}_{ji}) := 1.$$

This reduces our component by one supernode, and $\hat{r}$ remains connected and optimal. After iterating this procedure until all components are trivial, all supernodes contain exactly one node of range greater than 1, as demanded.

Now that we know this property of our optimal solutions, let us look what kind of graph $\hat{r}$ can induce. First note that the following implication holds:

$$\hat{r}(\hat{v}_{ij}) = 2 \implies \hat{r}(\hat{v}_{ji}) = 3$$

This means that every 2-supernode (defined analogously to 3-supernodes) has to be connected to a 3-supernode. This is clear, because if it were connected to a 2-supernode, these two supernodes would form a component not connected to the rest of the graph. (Recall here that we are in the case that an optimal range assignment includes a range 3 node.)

But now we know the structure of an optimal solution quite well: We have 3-supernodes connected in a clique-wise fashion, and each 3-supernode may have one 2-supernode attached to it. Figure 2.9 shows an example of such an optimal solution.

How can we optimize this kind of structure? This question is easy to answer: We want to use as many 2-supernodes as possible, so that we need the least number of 3-supernodes. In other words, we want to maximize the number of distinct pairs of 2- and 3-supernodes, connected by distance 2 edges, the edges of G. This means nothing else than constructing a maximum matching M in G, an efficiently solvable task.[Edm65, KV02] So here, we need additional cost $2|V| - |M|$ to establish connectivity.

Concluding our quite lengthy proof for our short remark, we give an algorithm to construct an optimal range assignment for $(1, 2, 3)$-Connectivity in $\hat{G}$.

- **Set** $\hat{r} = \mathbb{1}$.
- Construct a maximum matching M in G.
  Let $U \subseteq V$ be the set of nodes which are not covered by M.
- **If** $(|M| = 1)$,
  - construct an MST $T \subseteq G$.

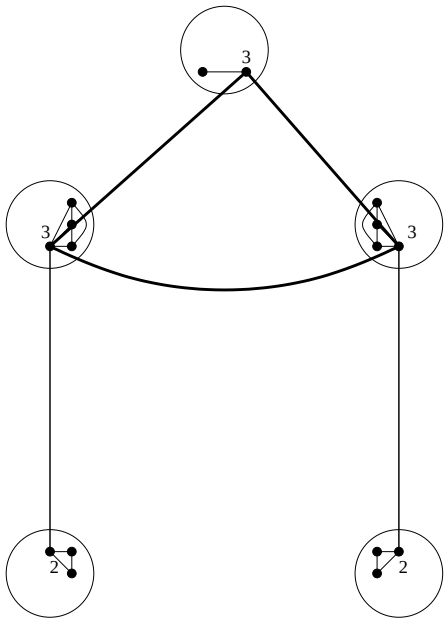

Figure 2.9: An optimal connected range assignment for the above example.

  - **For each** $\{v_i, v_j\} \in T$, **Set** $\hat{r}(\hat{v}_{ij}) = \hat{r}(\hat{v}_{ji}) = 2$.
  - **Output** $\hat{r}$ and **Stop**.

- **Else** ($|M| \geq 2$)
  - **For each** $\{v_i, v_j\} \in M$, **Set** $\hat{r}(\hat{v}_{ij}) = 3$ and $\hat{r}(\hat{v}_{ji}) = 2$.
  - **For each** $v_i \in U$, **Set** $\hat{r}(\hat{v}_{ij}) = 3$, for one arbitrary j.
  - **Output** $\hat{r}$ and **Stop**.

□

After this short remark (with its not so short proof) we turn to the question whether $(1,2,3)$-Connectivity also has a PTAS. It might not be as obvious as the one for $(1,2,3)$-Strong Connectivity, but one could try to design some more sophisticated approximation scheme for this special case, which has just enough structure to be NP-hard. We now note that we can modify our NP-hardness construction above slightly and get a proof of APX-hardness of $(1,2,3)$-Connectivity. This means that there cannot exist a PTAS for $(1,2,3)$-Connectivity, unless P = NP.

Theorem 2.13. *The* $(1,2,3)$-Connectivity *problem is APX-hard. More precisely, there cannot exist a* $(1+\frac{1}{1092})$*-approximation algorithm for* $(1,2,3)$-Connectivity*, or* $P = NP$.

Before we begin to prove this statement, let us first cite a fundamental non-approximability result on which all our APX-hardness proofs are based.

Lemma 2.14 (Chlebík and Chlebíková, [CC03]). *It is impossible to approximate:*

- *3-*Vertex Cover *to within* $1 + \frac{1}{99}$,
- *4-*Vertex Cover *to within* $1 + \frac{1}{52}$,
- *5-*Vertex Cover *to within* $1 + \frac{1}{50}$,

unless P = NP. This already holds for 3-, 4- resp. 5-regular graphs.

Proof (of Thm. 2.13): We show that the generic Set Cover reduction in the NP-hardness proof for $(1, 2, 3)$-Connectivity can be extended to an L-reduction from k-Vertex Cover, for an arbitrary fixed k. Recall that Vertex Cover is a special case of a Set Cover problem: we try to cover the edges of a graph with as few nodes as possible. Thus the edges are our elements, and the nodes are the subsets in this specific Set Cover problem. As edges are subsets of the nodes, this view point may be a bit counter-intuitive. This is reflected here by the fact that we would now have m nodes (subsets) and n edges (elements)[1]. As we feel this would lead to confusion and contradict conventional graph notation, let us better regard the problem as having n nodes (resp. subsets) and m edges (resp. elements).

Consider the construction in Theorem 2.7. We know from equation 2.1 that an optimal solution r^* for the reduced instance has cost

$$c(r^*) = \sum_{v_i \in V} r^*(v_i) = n + 2m + C^*, \tag{2.2}$$

where C^* is the size of a minimal set cover in the original instance. (We have omitted an additional cost of 1, because we can let all set nodes have distance 1, and the additional vertex u becomes redundant. It was only included to get a nicer drawing of the reduction in the first place.)

Together with some simple observations, we can get a bound on $c(r^*)$ linear in C^*. As we will also make use of these observations later on, let us formulate them as an intermediate lemma for future reference.

[1]This comes from the fact that Vertex Cover is literally not a covering problem but more of a Hitting Set problem. In a Hitting Set problem, we try to choose a minimum subset of the elements such that each subset is "hit" by at least one element. The Set Cover problem and the Hitting Set problem are completely identical in structure; it is more a matter of the preferred way of viewing at or describing the problem.

Lemma 2.15. *Let $G = (V, E)$ be a graph of maximum degree k, with n nodes and m edges. Let C^* denote the size of a minimum vertex cover in G.*
Then the two following facts hold:

$$C^* \geq \frac{1}{k} \cdot n \tag{2.3}$$

$$m \leq \frac{k}{2} \cdot n \tag{2.4}$$

These two inequalites directly imply

$$m \leq \frac{k^2}{2} \cdot C^* \tag{2.5}$$

Proof. Notice that in a graph of maximum degree k, each node can cover at most k edges, thus every vertex cover, in particular a minimal one, has to have size at least n/k.

As each node has at most k incident edges, and every edge has exactly 2 incident nodes, there cannot be more than $(k/2) \cdot n$ edges in total. □

Back the proof of Theorem 2.13. Assume there exists a $(1+\alpha)$-approximation algorithm $\mathcal{A}$ for $(1,2,3)$-Connectivity, for some constant $\alpha > 0$. Recall from Lemma 2.8 that we can easily obtain a vertex cover in the original instance G from the range assignment in the reduced Connectivity-instance. Let $r_{\mathcal{A}}$ be the range assignment constructed by algorithm $\mathcal{A}$, and $C_{\mathcal{A}}$ the vertex cover constructed from $r_{\mathcal{A}}$. Thus we have

$$c(r_{\mathcal{A}}) = n + 2m + C_{\mathcal{A}} \tag{2.6}$$

Because $\mathcal{A}$ is a $(1+\alpha)$-approximation, Lemma 2.15 gives us

$$\begin{aligned} c(r_{\mathcal{A}}) &= n + 2m + C_{\mathcal{A}} &&\leq (1+\alpha) \cdot (n + 2m + C^*) \\ \iff \qquad C_{\mathcal{A}} &&&\leq \alpha(n+2m) + (1+\alpha)C^* \\ &&&\leq \alpha(k+k^2)C^* + (1+\alpha)C^* \\ &&&= (1+\alpha(k^2+k+1))C^*. \end{aligned}$$

Thus, a polynomial $(1+\alpha)$-approximation algorithm for $(1,2,3)$-Connectivity would automatically imply the existence of a polynomial $(1+\alpha(k^2+k+1))$-approximation algorithm for k-Vertex Cover. We would therefore have

- a $(1+13\alpha)$-approximation algorithm for 3-Vertex Cover,
- a $(1+21\alpha)$-approximation algorithm for 4-Vertex Cover and
- a $(1+31\alpha)$-approximation algorithm for 5-Vertex Cover.

Combining this with Lemma 2.14, we see that

- $\alpha \geq \frac{1}{13 \cdot 99} = \frac{1}{1287}$ must hold for 3-Vertex Cover,
- $\alpha \geq \frac{1}{21 \cdot 52} = \frac{1}{1092}$ for 4-Vertex Cover and
- $\alpha \geq \frac{1}{31 \cdot 50} = \frac{1}{1550}$ for 5-Vertex Cover,

or P = NP. We notice that our reduction combined the hardness result delivers the best result for 4-Vertex Cover, proving our statement. □

As some more APX-hardness proofs with explicit non-approximability results are coming up, we have given this one in maybe more detail than absolutely necessary so that the outline becomes clear. Later proofs will be shorter.

Of course, the inapproximability constants depend on the actual values that distances may take. We can get better results with other values:

Corollary 2.16. *There cannot exist a* $(1 + \frac{1}{468})$*-approximation algorithm for* $(0, 1, 2)$-Connectivity*, or P = NP.*

Proof. In the construction to prove Theorem 2.13, we can reduce each distance by 1 to get an APX-hardness proof for $(0, 1, 2)$-Connectivity. With these distances, Equation 2.6 becomes

$$c(r_{\mathcal{A}}) = m + C_{\mathcal{A}}$$

and we get

$$\begin{aligned} c(r_{\mathcal{A}}) &= m + C_{\mathcal{A}} \leq (1 + \alpha) \cdot (m + C^*) \\ \Longrightarrow \quad C_{\mathcal{A}} &\leq (1 + \alpha(\frac{k^2}{2} + 1))C^* \end{aligned}$$

With this, we can calculate that

$$\alpha \geq \frac{1}{9 \cdot 52} = \frac{1}{468}$$

must hold (mod. P = NP) because of the inapproximability result for 4-Vertex Cover. □

With $(1, 2, 3)$-Connectivity being APX-hard and $(1, 2, 3)$-Strong Connectivity allowing a PTAS, does Strong Connectivity maybe allow a PTAS in general? To see it does not, consider the original construction in Theorem 2.2 again, which works for $(1, 2, \infty)$-Strong Connectivity. In this setting, there is no alternative to the bidirected links as used in the Connectivity version of the problem. Thus radii increased on top of a minimal configuration again directly correspond to a set cover in the original instance, and the proof for Theorem 2.13 works in the same way for $(1, 2, \infty)$-Strong Connectivity. The same of course also holds for $(1, 2, \infty)$-Connectivity.

Corollary 2.17. *There cannot exist a* $(1 + \frac{1}{1092})$*-approximation algorithm for neither* $(1, 2, \infty)$-CONNECTIVITY *nor* $(1, 2, \infty)$-STRONG CONNECTIVITY*, unless* $P = NP$*.*

So $(1, 2, \infty)$-STRONG CONNECTIVITY is APX-hard, while $(1, 2, 3)$-STRONG CONNECTIVITY allows a PTAS. A question is what happens "in between", i.e., is there a constant k for which $(1, 2, k)$-STRONG CONNECTIVITY has a PTAS, while $(1, 2, k + 1)$-STRONG CONNECTIVITY is APX-hard? This is not the case; $(1, 2, k)$-STRONG CONNECTIVITY admits a PTAS for every constant k. In fact, the following statement can be shown.

Theorem 2.18. $(1, 2, \ldots, \Delta - 1, \Delta)$-STRONG CONNECTIVITY*admits a PTAS. More generally, let* $\delta > 0$ *be the smallest non-zero distance occuring in some range assignment problem, and* Δ *the largest distance.* STRONG CONNECTIVITY *on instances where the quotient* $\frac{\Delta}{\delta}$ *is bounded by a constant admits a PTAS.*

Proof. The proof is similar to the one of Corollary 2.11. We can assume wlog. that $\delta = 1$, else we divide everything by δ. We know that a feasible range assignment for STRONG CONNECTIVITY has cost at least mst. On the other hand, as before, there exists a feasible range assignment of cost $mst + \Delta$: Direct an MST towards some node v and set the radius of v to Δ to transmit data to all other nodes. We call the resulting range assignment the 'all-to-one, one-to-all' solution. This range assignment obviously is a $(1 + \frac{k}{mst})$-approximation. In case $k/mst \leq \varepsilon$, we are done. Otherwise, $k/mst > \varepsilon$ implies $mst < k/\varepsilon = O(1)$, thus $n = O(1)$. Every node has $n-1$ possible choices of distances for its radius (it can transmit to $1, 2, \ldots$ or $n-1$ other nodes), making $(n-1)^n = O(1)$ possible range assignments in total, which we can enumerate in constant time. □

2.2 Results for BROADCAST

The hardness results for BROADCAST are also all based on very generic reductions from SET COVER. We can use nearly the same reduction as in Theorem 2.2, only slightly easier, for the graph version of BROADCAST. It already works for the unweighted version of this problem, i.e., we seek a range assignment with the least number of nodes having one or more outgoing edge. In the terms we introduced last section, this means that already $(1, \infty)$-BROADCAST is very hard.

Theorem 2.19. $(1, \infty)$-BROADCAST *is as hard as the* SET COVER *problem. I.e., it is NP-hard, and moreover, there cannot be an approximation better than* $O(\log n)$*, or* $P = NP$*.*

Proof. Take the construction of the proof for Theorem 2.2, but set the distance on *all* edges to 1. The node u is the root node which has to send data to all other

nodes. $r(u) = 1$ will hold of course, so all the set nodes are reached already. In particular, it does not make sense to have any element node with $r(v_i) \neq 0$.

Now all the element nodes have to be reached by using as few set nodes with $r(w_j) = 1$ as possible. This corresponds directly to the original Set Cover problem: The non-zero set nodes w_j in an optimal range assignment are exactly the covering sets S_j in an optimal set cover. Because of $r(u) = 1$, the cost of an optimal range assignment r^* amounts to

$$c(r^*) = 1 + C^*,$$

where C^* is the size of an optimal set cover. Thus, an inapproximability result for Set Cover holds for $(1, \infty)$-Broadcast as well. □

So Broadcast is very hard already in a very generic setting. This result means that it does not make any sense to try a constant factor approximation for this problem. However, in some special settings, the problem becomes easy, or even trivial. As in the last section, we see what happens if Δ/δ, the quotient of the largest and smallest non-zero distance, is bounded by a constant. Recall that Connectivity is APX-hard while Strong Connectivity admits a PTAS but remains NP-hard.

Theorem 2.20. *Let $\delta > 0$ be the smallest non-zero distance occuring in some range assignment problem, and Δ the largest distance.* Broadcast *on instances where the quotient $\frac{\Delta}{\delta}$ is bounded by a constant is polynomially solvable.*

Proof. As before, assume wlog. that $\delta = 1$. Let s be the broadcasting source node. The range assignment

$$r(v) = \begin{cases} \Delta & \text{for } v = s, \\ 0 & \text{else.} \end{cases}$$

is a feasible broadcast scheme. We only need to investigate cheaper schemes. Such a range assignments has at most Δ nodes with non-zero power consumption. Thus, when we try every subset of size at most Δ, and try every of the possible n values (transmitting to $0, 1, \ldots$ or $n-1$ other nodes) for those (at most) Δ nodes, we have tried out every cheaper possible range assignment, and have found the optimum in the end. We "only" need to check

$$\binom{n}{\Delta} \cdot n^{\Delta} \leq n^{2\Delta}$$

possible range assignment, a polynomial number. □

So while (Strong) Connectivity remains hard under bounded distances, Broadcast becomes polynomial in this setting. We now come to a very important setting where Broadcast is completely trival.

Remark 2.21. For distances satisfying the triangle inequality, BROADCAST is trivial.

Proof. It is always possible to have source node s transmit directly to each node. I.e., when

$$\Delta_s := \max_{v \in V} \; d(s, v),$$

range assignment

$$\hat{r}(v) = \begin{cases} \Delta_s & \text{for } v = s, \\ 0 & \text{else.} \end{cases}$$

is feasible. Obviously, $c(\hat{r}) = \Delta_s$. On the other hand, in any feasible broadcast scheme r, we have to reach every node from s. So r costs at least

$$c(r) \;\geq\; \max_{v \in V} \sum_{e \in P(s,v)} d(e),$$

where $P(v, w)$ is a shortest path from v to w. With the triangle inequality, we know that

$$c(\hat{r}) = \Delta_s \leq \max_{v \in V} \sum_{e \in P(s,v)} d(e) \leq c(r)$$

for every feasible range assignment r, so $\hat{r}$ is feasible with least possible cost. □

E.g., in the reduction for SET COVER-hardness of BROADCAST, when the $\triangle$-inequality would hold, node u could send everywhere with just a radius of 2.

Recall that in geometric range assignment problems, network distances are Euclidean distances in some space to the power of the constant power-distance gradient α. Note that for $\alpha > 1$ the distances in this kind of model do not satisfy the $\triangle$-inequality, but a weaker inequality, dependent on α. We call this the $\triangle^\alpha$-inequality.

Definition 2.22. We say that distances $d : V \times V \to \mathbb{R}_+$ satisfy the $\triangle^\alpha$- inequality, if for every three points $u, v, w \in V$, we have that

$$d(v, w) \;\leq\; \left(\sqrt[\alpha]{d(v, u)} + \sqrt[\alpha]{d(u, w)} \right)^\alpha$$

Correspondingly, we call $d : V \times V \to \mathbb{R}_+$ an *α-metric* if the following conditions hold:

1. $d(v, u) = 0 \iff v = u$
2. $d(v, u) \;=\; d(u, v)$
3. d satisfies the $\triangle^\alpha$-inquality.

The distance functions in our geometric instances are α-metrics as the $\triangle^\alpha$-inequality holds, because the $\triangle$-inequality holds for $\|.\|$, and $d(v,w) = \|v-w\|^\alpha$.

As is mentioned later, there exist constant factor approximation algorithms for geometric instance of Broadcast in $\mathbb{R}^d$ for constant dimension d, and constant power-distance gradient α. One possibility to design such an approximation for general geometric instances might be to formulate one for general instances satisfying the $\triangle^\alpha$-inequality. One would also overcome the sometimes tedious use of geometry, if the $\triangle^\alpha$-inequality would already embrace the necessary structure to allow a constant factor approximation.

Indeed, the instances constructed in the proof of Theorem 2.19 become trivial when we demand the $\triangle^\alpha$-inequality. In this case, the source node r cannot have a distance larger than 2^α to any element node, a constant. So by Theorem 2.20, we can solve these instances in polynomial time.

In order to fix this construction, one should, in some way, "separate" distinct set nodes from each other, so that no single set node w_j can cover more element nodes than $\{v_i \mid i \in S_j\}$ at "small" extra cost. This is of course possible, but maybe only at forbiddingly large overhead costs, losing the property of being approximation preserving.

But this is not the case. We give a construction which spreads out the nodes, virtually without adding any extra costs. Although the number of nodes increases dramatically, the reduction is still polynomial.

Theorem 2.23. Broadcast *with* $\triangle^\alpha$ *is* Set Cover*-hard, i.e., there cannot be an approximation better than* $O(\log n)$*, or* $P = NP$*.*

Proof. We present our construction in three steps. First, we design an auxiliary graph G similar to the one in the construction for $(1,\infty)$-Broadcast. Then, certain edges in G are replaced by lines of very many very close points, ensuring approximation preservation. The resulting graph is called G'. Finally, from G' we design $\hat{G}'$, which has a distance matrix satisfying the $\triangle^\alpha$-inequality and proving our claim.
We first define how we get from G' to $\hat{G}'$:

Definition 2.24. Let $G = (V, E, w)$ be a graph with a weight function $w : E \to \mathbb{R}_+$ on its edges. Let $\hat{G} = (V, V \times V, d)$ where $d : V \times V \to \mathbb{R}_+$ is defined as follows:

$$d(v,w) \;=\; |P(v,w)|^\alpha,$$

where $P(v,w)$ is a shortest path in G between v and w.

Directly from the definition of $\hat{G}$ we see that d is an α-metric. So we call d (or $\hat{G}$) the *α-metric induced by G*. Obviously, the 1-metric induced by G is the standard metric induced by G.

Now to the step from G to G'. We want to reduce the cost of certain edges which we need as overhead, but we would like to pay least possible in our cost

function, as they interfere with the approximation preservation. The following Lemma says we can get them virtually for free.

Lemma 2.25. *Suppose in a graph G we have edges of total length L. We replace these edges by a line of N stations at distance $1/N$ per unit weight, and regard the α-metric graph $\hat{G}$. For constant $\alpha > 1$, we can choose N to be a polynomial in L such that the total power cost in $\hat{G}$ to have each line segment (strongly) connected is less than 1.*

Proof. When we put N stations on a line of length 1, this line gets (strongly) connected when each station transmits with radius $1/N$. The total power cost on this line is thus $N \cdot \left(\frac{1}{N}\right)^{\alpha}$. So if we want to have the power used on all lines to be below 1,

$$LN \cdot \left(\frac{1}{N}\right)^{\alpha} \;<\; 1$$

must hold. For $\alpha > 1$, this is equivalent to $N > L^{\frac{1}{\alpha-1}}$, a polynomial in L. □

Note that the step from G to $\hat{G}$ is not necessarily polynomial as length L might be encoded in size $\log(L)$. However, it will be polynomial in the input size in our reduction.

After these preparations, we can begin with our construction. Suppose we are given an instance of Set Cover with sets $S_1, \ldots, S_m$ and elements $1, \ldots, n$. We first construct $G = (V, E, w)$ from this instance.

A difference to the constructions above is that we now introduce an element variable v_i^j for each set S_j element i appears in: Let

$$V = \{u, w_1, \ldots, w_m, \} \cup \{v_i^j \mid i \in S_j\}.$$

We describe edge set E by two distinct edge sets $E_1 \cup E_n = E$. The set $E_1 = \{\{w_j, v_i^j\} \mid i \in S_j\}$ represents the set/element relation. Set E_n, containing the overhead in this reduction, is defined as

$$E_n = \{\{u, w_j\} \mid j = 1, \ldots, m\} \cup \{\{v_i^j, v_i^{j'}\} \mid \{j, j'\} \in J_i\},$$

where $J_i = \{\{j, j'\} \mid j < j', i \in S_j, i \in S_{j'} \text{ and } i \notin S_{\tilde{\jmath}}\ \forall_{j<\tilde{\jmath}<j'}\}$. So E_n contains edges between the root and set vertices, and connects appearences of the same element i in different sets along the path J_i. We finish our description of G by setting $w(e) = 1$ for all $e \in E_1$, and $w(e) = n$ for $e \in E_n$. Figure 2.10 shows an illustration of this reduction.

Remark 2.26. $|P(x, y)| \leq 2$ holds only if x and y are element or set nodes of the same set S_j; otherwise, $|P(x, y)| \geq n$ holds.

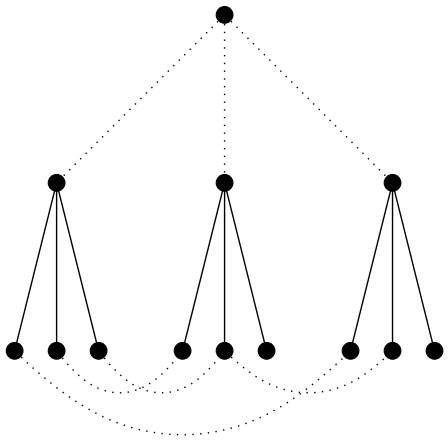

Figure 2.10: Graph G for an example Set Cover instance. Edges in E_n are dotted.

We have thus spread out our old reduction, but at the cost of high weights on the overhead edges in E_n. We are now going to reduce these weights with the help of Lemma 2.25, and call the resulting graph G'.

First we fix a number N which we calculate later. For each edge $e = \{x, y\} \in E_n$ (the edges of length n), we do the following. We remove edge e from the graph, insert a line of $n \cdot N$ new vertices $\{l_1^e, \ldots, l_{nN}^e\}$ with length $1/N$ edges $\{\{l_i^e, l_{i+1}^e\} \mid i = 1, \ldots, nN - 1\}$ into the graph, and connect l_1^e to x, and l_{nN}^e to y. We call the resulting graph G'. In the example in Figure 2.10, one could say the dotted edges have now become dotted indeed.

We have m edges $\{u, w_j\} \in E_n$. For a single element i appearing in at most all m sets, the path connecting its occurrences $\{v_i^j\}$ in E_n has at most $(m-1)$ edges. Thus, all paths in total contain no more than $(m-1)n$ edges. So number L in Lemma 2.25 would here be

$$L = (m-1)n^2 + mn,$$

and we choose $N = L^{\frac{1}{\alpha-1}}$. Note that N (and also LN) is a polynomial in the input size of our original Set Cover instance, so G' has a polynomial number of nodes, and our entire reduction is polynomial.

Now regard $\hat{G'}$, the α-metric induced by G'. In a minimal configuration for $\hat{G'}$, each node has radius $(1/N)^\alpha$, at negligible total cost. As in the reduction for $(1, \infty)$-Broadcast, one way to broadcast everywhere from u is to raise the radii of the set nodes corresponding to a set cover in the original instance to 1. The cheapest alternative to cover elements in different sets is to set the range in u to $(n+1)^\alpha > n + 1$, which is forbiddingly large. So, the cost of an optimal range assignment is equal to the size of an optimal set cover (plus 1, which is negligible).

Thus, any approximation for BROADCAST better than $O(\log n)$ would imply such an approximation for SET COVER, and P = NP.

□

It is not surprising that the number of nodes explodes in this reduction for $\alpha \to 1$. It is remarkable that this problem is completely trivial for $\alpha \leq 1$, yet not constant factor approximable for any $\alpha > 1$.

Furthermore, the identical reduction shows that STRONG CONNECTIVITY is APX-hard. To get a linear reduction from an APX-complete problem, we apply it on a special class of SET COVER instances, the VERTEX COVER problem.

Corollary 2.27. *The* STRONG CONNECTIVITY *problem with $\triangle^\alpha$-inequality is APX-hard for any $\alpha > 1$. More precisely, there cannot exist a $(1 + \frac{1}{468})$-approximation algorithm for this problem, unless P = NP.*

Proof. In a minimal configuration, all nodes have radius $(1/N)^\alpha$. The resulting connectivity graph has one component containing all set nodes, and one for each element i, containing all its occurrences. Each of the latter components needs to send out of this component, which it can do by setting any of the occurrences to radius 1. Sending any further than this would not bring any advantage:

At radius 2^α, node v_i^j could send to the other nodes $v_{i'}^j$ for which $i' \in S_j$. But it is not only cheaper to set $r(v_i^j) = r(w_j) = 1$ instead. Moreover, this allows to send from the set node component into the i'-components for which $i' \in S_j$. Any other possibility to increase connectivity by setting a radius to more than 1 is forbiddingly expensive, as we know thanks to Remark 2.26.

Thus, an optimal strongly connected range assignment will have exactly one radius 1 node v_i^j per element component i (at an arbitrary j). All components now send into the set component, so it suffices to have the set component send into all element components. This can be done by setting the radii of the set nodes of a set cover to 1. All other nodes have neglectable radius $(1/N)^\alpha$. In total, a cheapest feasible range assignment costs less than

$$m + C^* + 1,$$

where C^* is the size of a smallest set cover. This is the same amount as in Corollary 2.16, so we get the same inapproximability result by reducing from 4-VERTEX COVER. □

It remains unclear what happens for $\alpha \leq 1$, i.e., the STRONG CONNECTIVITY problem with $\triangle$-inequality. We have seen PTASs for various cases, and we know it is APX-hard in the general case, also with the $\triangle^\alpha$-inequality. Its NP-hardness proof with distances $(1, 2, 3)$ was also the only reduction not from a SET COVER type problem, namely from HAMILTONIAN CYCLE. Although already $(1, 2)$-TSP is APX-hard [PY93], it is not clear how to get an L-reduction to metric STRONG

Connectivity. The problem with the construction in the prior Corollary is that we cannot use Lemma 2.25 any longer to control the overhead size: We have $O(n)$ edges of length n, contributing cost $O(n^2)$ to the cost function. Thus, we do not have an L-reduction any longer. But with less overhead, it is unclear how to rule out that the 'one-to-all, all-to-one'-solution becomes cheaper than a set cover solution for growing instance size at some point.

2.3 Overview and conclusion

We summarize the results of this chapter and discuss directions for future research.

	Connectivity	Strong Con.	Broadcast
unweighted Graphs	trivial	trivial	Set Cover hard
Graphs with weights 1 and 2	APX-hard	APX-hard	Set Cover hard
$(1,2)$-version	polynomial	polynomial	trivial
$(1,2,3)$-version	APX-hard	NP-hard; PTAS	trivial
$\triangle$-inequality	APX-hard	NP-hard	trivial
$\triangle^{\alpha}$-inequality	APX-hard	APX-hard	Set Cover hard

Table 2.1: Summary of hardness results in this chapter.

Table 2.1 summarizes the main complexity results of this chapter. Concerning unweighted graphs, strictly speaking we should say graphs where all edges have the same weight, maybe 1. The row for $(1,2,3)$-problems generalizes to settings where the quotient between largest and smallest distance is bounded by a constant. However, Broadcast is not really trivial in this setting, but solving it by brute force becomes (theoretically) polynomial. It is similar to the PTAS for its bounded Strong Connectivity counterpart.

It is interesting to see the jump complexity-wise in the graph version of (Strong) Connectivity when we allow two instead of one weight. The difference between (Strong) Connectivity on one side and Broadcast on the other is also apparent. In contrast to other optimization problems like TSP and Steiner Tree, (Strong) Connectivity remains polynomial when restricted to its $(1,2)$-versions, and becomes hard when we allow one more choice for the weights. The $(1,2,3)$-version of the problems is very interesting as the three problems fall into three different approximation classes. It reveals the difference between Connectivity and its directed counterpart, two quite similar problems.

All in all, the characterization of the three problems in the various restricted settings is nearly complete. What remains an interesting open problem is the

Strong Connectivity problem with $\triangle$-inequality. NP-hardness is shown already for $(1,2,3)$-Strong Connectivity with $\triangle$-inequality, and there is a PTAS when the quotient of longest and shortest distance is bounded by a constant. Yet it is still unclear whether there exists a PTAS also for general metric Strong Connectivity, whether it is APX-hard or whether none of both applies. Due to the strong similarity to Connectivity, the author's intuition was that metric Strong Connectivity should be APX-hard as well. However, the case seems not so clear any more. The 'all-to-one, one-to-all' solution is made attractively cheap by the $\triangle$-inequality, and has so far destroyed all reduction attempts. When one tries to 'spread out' the reduction as for the $\triangle^\alpha$-inequality, it is not clear how to do this without destroying its linearity. On the other hand, it is also unclear how to get a general PTAS. The 'one-to-all, all-to-one' solution alone is not enough; it can be as big as 1.5 times the optimum, as we will see later. However, it will yield a PTAS for some special geometric instances.

As a final remark, relaxing the $\triangle$-inequality to the $\triangle^\alpha$-inequality makes the problems as hard as their general version, for any $\alpha > 1$. In particular, we cannot hope for a constant factor approximation for Broadcast as in the geometric versions by just extracting this special feature. This observation corresponds with [CCP+01b] where the constant factor approximations for Broadcast are only constant for fixed dimension; in fact, they grow exponentially with the dimension.

Chapter 3

Complexity of geometrical instances

In the preceeding chapter, we have proved computational hardness for (Strong) Connectivity and Broadcast in various settings. As these problems are motivated by real-world applications, it is interesting to know if instances which model real-world situations are still hard to solve, or if they get considerably easier. We concentrate on range assignment instances which are defined by points in 2- and 3-dimensional space, and a cost function depending on the Euclidean distance between two points that model the power consumption for transmitting data across this distance.

Definition 3.1 (Geometric Range Assignment problems).

Instance: A set of points $S \subseteq \mathbb{R}^d$ in d-dimensional space, a constant "power-distance gradient" $\alpha > 0$, and a certain network property Π.

Task: Find a network $F \subseteq E$ satisfying network property Π which minimizes the cost function

$$c(F) = \sum_{v \in V} \max_{(v,w) \in F} \|v - w\|^\alpha,$$

where $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^d$.

Geometric Range Assignment problems are special cases of Range Assignment problems as defined in the previous chapter: We identify the points in $S = \{s_1, s_2, \ldots, s_n\}$ with a set $V = \{1, \ldots, n\}$, set $E = V \times V$ and define $d(i,j) = \|s_i - s_j\|^\alpha$, and get an equivalent instance $G = (V, E, d)$ for a Range Assignment problem. As this whole chapter is about geometric instances, we will omit the word "geometric" from the problem description.

At this point, we again stress a certain ambiguity about terms. We regard $d(i,j)$ as the "distance" between points s_i and s_j, which is actually the Euclidean

distance to the power of α. So keep in mind that when we talk about distances between points, this term is only related to their natural distance, but not the same.

In [CPS04], the authors have tried to capture what makes out a 'realistic' 2-d instance via the concept of *well-spread* instances. They give an approximation algorithm for this kind of instances for range assignment problems with bounded hops, which is however not the focus of this thesis. We define this concept as in [CPS04] and give a natural generalization for higher dimensions. Define

$$\begin{aligned} \Delta(S) &= \max\{\|u-v\| \mid u,v \in S\} \\ \delta_s(S) &= \min\{\|s,v\| \mid v \in S \setminus \{s\}\} \\ \delta(S) &= \min\{\delta_s(S) \mid s \in S\} \end{aligned}$$

Definition 3.2. As in [CPS04], we say that a family $\mathcal{S}$ of 2-dimensional instances is *well-spread* if there exists some positive constant c such that, for any $S \in \mathcal{S}$, $\delta(S) \geq c\Delta(S)/\sqrt{|S|}$ holds. A natural generalization for other dimensions is to call a family $\mathcal{S}$ of d-dimensional instances well-spread if there exists some positive constant c such that, for any $S \in \mathcal{S}$,

$$\delta(S) \geq c\Delta(S)/\sqrt[d]{|S|}$$

holds.

Orthogonal regular grids of full dimension are the prototypical well-spread instances. In the following, we sometimes omit the specific set of stations S if it is clear from the context which S is meant.

These problems have of course a very special structure. We have seen that metric instances resp. α-metric instances are just as hard as in the general case (except for metric Broadcast), but on the other hand, the reductions are not realizable in a geometric setting. So we have to make some extra effort, which this chapter is about.

3.1 Previous work and our results

The complexity of geometric range assignment problems has been investigated before. The first result in this area is by Kirousis *et al.* [KKKP00], who have shown that Strong Connectivity is NP-hard in $\mathbb{R}^3$ for $\alpha \geq 1$. They also presented a dynamic program that solves Strong Connectivity on the real line $\mathbb{R}$ to optimality in time $O(n^4)$. Clementi, Penna and Silvestri [CPS04] proved that Strong Connectivity with $\alpha \geq 2$ in $\mathbb{R}^2$ is NP-hard, and even APX-hard in $\mathbb{R}^3$. These constructions were adapted in [CCP+01b] to prove NP-hardness of Broadcast in 2-d.

The survey [CHP+02] gives a good coverage of these and other results in this area, and raises questions for further research. One open problem stated therein is the case of (STRONG) CONNECTIVITY with $\alpha = 1$, i.e., Euclidean distances. It is conjectured in [CHP+02] »*... that this case is efficiently solvable or, at least, approximable within a better factor than 2.*« Another open question in this survey is whether BROADCAST allows a PTAS. These two questions are answered in this thesis. It can be shown that (STRONG) CONNECTIVITY remains NP-hard for $\alpha = 1$ (in fact, for any constant $\alpha > 0$) in $\mathbb{R}^2$. Furthermore, the first overall APX-hardness proof for BROADCAST is presented.

These results are mostly due to new, more simple and flexible reductions for our problems. The disadvantage of the existing reductions is that they consist of technically quite involved gadgets which cannot be adapted as easily to work for similar range assignment problems. The higher versatility of our reductions is underlined by improved inapproximability results where APX-hardness was known before. In addition, we give the first hardness results for well-spread instances.

3.2 Outline of the generic reduction

The structure of our reduction is to a great part inspired by the classic reduction for NP-hard of the RECTILINEAR STEINER TREE PROBLEM by Garey and Johnson [GJ77]. We therefore outline their proof as much as we need to make this chapter self-contained.

Given a finite set of points V lying in the real plane $\mathbb{R}^2$, the RECTILINEAR STEINER TREE PROBLEM seeks to find a tree interconnecting V using only horizontal and vertical lines of shortest possible total length. The reduction in [GJ77] starts from PLANAR VERTEX COVER, which was shown to be NP-hard one year earlier in [GJS76]. Remarkably, in [GJ77], as a by-product NP-hardness of PLANAR 3-VERTEX COVER is proven on the way. (Of course, 2-VERTEX COVER is trivial.)

From now on, we will often abbreviate VERTEX COVER as VC, and k-VC for k bounded degree VERTEX COVER. The line of reductions in [GJ77] is as follows:

> PLANAR VC → PLANAR 3-VC → PLANAR CONNECTED 4-VC → RECTILINEAR STEINER TREE PROBLEM

where a connected vertex cover is a vertex cover whose node set induces a connected graph.

3.2.1 The backbone

The step from PLANAR 3-VC to PLANAR CONNECTED 4-VC is of particular interest for our purposes as it builds the bridge from a covering problem like

Vertex Cover to a network problem. See Figures 3.1 and 3.2 for a graph with distinct vertex cover and connected vertex cover.

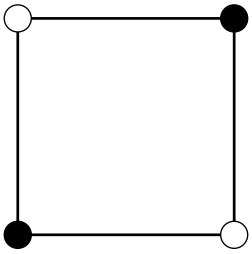

Figure 3.1: A graph with a vertex cover, indicated by black nodes ...

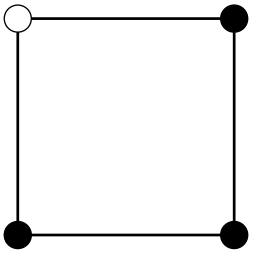

Figure 3.2: ... and the same graph with a connected vertex cover.

Roughly said, a kind of "backbone" structure is added to the Planar 3-VC instance. This backbone ensures that vertices of the original instance are not only chosen in order to maintain connectivity when they are unnecessary for covering. Instead, connectivity is always ensured by the construction of the backbone.

Maybe this construction is best explained by a picture. Figures 3.3 and 3.4, taken from [GJ77], show a planar drawing D of an example instance of 3-VC and our reduced Connected 4-VC instance $\bar{D}$, which is almost but not entirely the same as D_{GJ}, which we call the graph constructed in [GJ77]. $\bar{D}$ is constructed in the following way (for a rigorous proof, the reader is of course invited to refer to the original proof in [GJ77]):
Let $D = (V, E)$ be a planar graph with maximum degree 3 with a fixed planar embedding, and let $n = |V|$ and $m = |E|$ be the number of vertices resp. edges of the original VC-instance D.

- First, split each edge $e = \{x, y\} \in E$ into three edges $\{x, x_e\}$, $\{x_e, y_e\}$ and $\{y_e, y\}$ by adding two new vertices x_e, y_e per edge. Call those new vertices

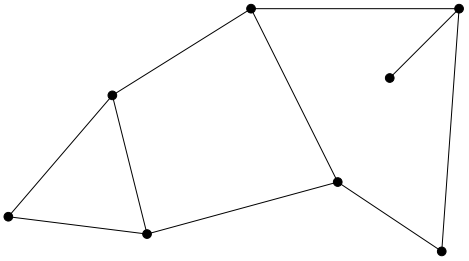

Figure 3.3: An instance D of 3-VC...

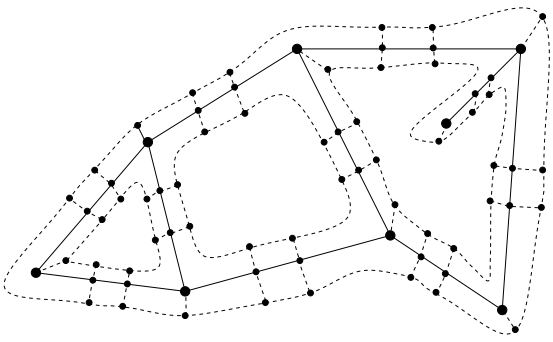

Figure 3.4: ... and the reduced Connected 4-VC instance $\bar{D}$. Backbone edges are dashed.

C ("connectors") and the split edges E'. We call this intermediate graph $D' = (V \cup C, E')$.

- For each vertex $c \in C$, place one new vertex $b_{c,R}$ in each adjacent region R (one or two), and connect $b_{c,R}$ to c. For each vertex $v \in V$ of the original graph, place one new vertex $b_{v,R}$ in any neighboring region R, and connect $b_{v,R}$ to v.
- In each region R, connect all vertices $b_{\cdot,R}$ by a walk along the border of the region, like in Figure 3.4. Collect the additional edges of this and the previous step in the set $\bar{E}$, and the nodes in the set B ("backbone").

This completes the construction of the planar graph $\bar{D} = (V \cup C \cup B, E' \cup \bar{E})$ with a fixed embedding. Note that the connector nodes are only needed to keep the backbone connected whilst maintaining planarity. Of course they must also yield a correct reduction. This is shown by the following lemma, which is implicit in the construction of Garey and Johnson.

Lemma 3.3. *D has a vertex cover of size $k \Leftrightarrow D'$ has a vertex cover of size $k + m$.*

Proof. Let N be a vertex cover of D. Let $M = \{x_e \mid e = \{x, y\} \in E,\ y \in N\} \cup \{y_e \mid e = \{x, y\} \in E,\ y \notin N\}$. Now $|M| = m$, and $N \dot\cup M$ is a vertex cover of D'.
Conversely, let N' be a vertex cover of D', and let $M' = \{x \mid e = \{x, y\} \in E,\ x_e, y_e \in N'\}$. Now $N = N' \cap V \cup M'$ is a vertex cover of D, and has at least m nodes less than N'. □

The constructed graph $\bar{D}$ that we present here is in fact slightly different from the graph D_{GJ} constructed in [GJ77]: In D_{GJ}, each backbone node $b \in B$ has a "spike", i.e., it is additionally connected to a copy of itself which has no other neighbor. The single purpose of these spikes is to ensure that, wlog., the whole set of backbone nodes B is included in every vertex cover of $\bar{D}$. Additionally, one connector per edge in D must lie in every vertex cover, providing connectivity of the backbone. As every node has a backbone neighbor, all feasible vertex covers for $\bar{D}$ are wlog. connected. So the size of a minimum vertex cover for D is k iff a minimum connected vertex cover of D_{GJ} has size $k + m + |B|$.

In the context of range assignments for radio stations, we will not need these spikes to ensure that the backbone is connected.

3.2.2 Graph drawing

In the next step, an *orthogonal drawing* of $\bar{D}$ in $\mathbb{R}^2$ is needed. Efficient methods for this task have been already known and used by Garey and Johnson in their Rectilinear Steiner Tree reduction.

In an *orthogonal drawing* of a graph, each edge is represented by an axis-parallel line or by several adjacent line segments. In the latter case, the edge has a "bend" in the drawing. Crossings of horizontal and vertical lines other than bends (thus, of degree 3 or higher) must correspond to a node. All nodes and bends have integer coordinates.

Obviously, a graph must be planar and of maximal degree 4 to have an orthogonal drawing in the plane and of maximal degree 6 to have an orthogonal drawing in 3-d space. It is a classical result in the field of graph drawing that these necessary conditions are also sufficient.

Lemma 3.4. *It is possible to efficiently construct orthogonal drawings of planar graphs with maximum degree 4 in 2-d and arbitrary graphs with maximum degree 6 in 3-d with maximum edge-length $O(n)$.*

See [ESW96] for a reference in this huge field, where even an orthogonal drawing in 3-d with edge length $O(\sqrt{n})$ is constructed.

3.2.3 Placing the stations

We will finally get our RANGE ASSIGNMENT instances by replacing the lines of the drawing by equidistant stations, with (possibly) some free space at the ends. The actual way of doing this differs from problem to problem. In the 2-d constructions, an important ingredient is to take different distances on backbone and original edges, while in the 3-d constructions (which need to be approximation preserving, but have a simpler backbone construction), the free space around original nodes distinguishes backbone lines and original lines.

3.3 Hardness results for CONNECTIVITY

We first present our reductions for CONNECTIVITY, because they appear here in their most generic form.

3.3.1 NP-hardness of CONNECTIVITY in 2-d

Every reduction in 2-d starts out with graph $\bar{D}$ as constructed in the prior section. We shortly describe the construction of the final CONNECTIVITY instance out of $\bar{D}$.

- Construct an orthogonal drawing of $\bar{D}$ in the plane.
- Scale the whole drawing by the factor 3.
- Replace each line in the drawing by a set of equidistant points in the following way: Place one station at one end of the line, and:

– For each polyline representing an edge originally in E', place stations on every point with integer coordinates, i.e. points at distance one.

– For each line representing some part of a backbone edge (those in $\bar{E}$), place stations at distance $\frac{3}{4}$.

By scaling by a factor c we mean multiplying the coordinates of each point with c. By a *vertex (or node) station* we mean a station representing a node of the graph D'. By an *edge-end* we mean the last station on an edge before the vertex station.

Note that because of the scaling step, the first and last station on a straight line segment always have integer coordinates, and the minimum distance between two vertex stations is 3. See figures 3.5 and 3.6 for an illustration of this reduction.

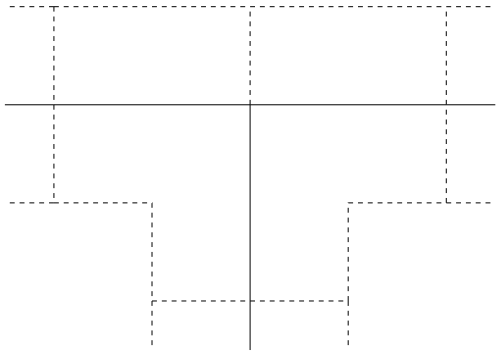

Figure 3.5: A small part of an orthogonal drawing of a graph $\bar{D}$.

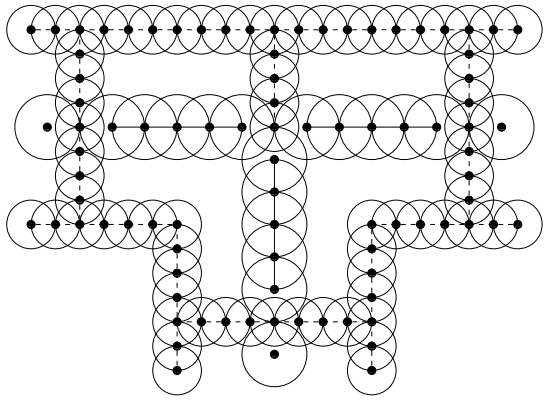

Figure 3.6: The resulting set of stations, with a minimal configuration and the induced communication graph.

Originally given a planar instance D of 3-Vertex Cover with a fixed embedding, we have constructed a blown-up version $\bar{D}$ and, by our last step, associated a set of points S in the plane with D. We now claim that a solution for the CONNECTIVITY problem for S also automatically yields a minimal vertex cover for D.

Theorem 3.5. *For any $\alpha > 0$,* CONNECTIVITY *in $\mathbb{R}^2$ is NP-hard.*

Proof. Let us look at the minimal configuration r_{min} for S: All stations on edges in E' have $r_{min} = 1$, and all stations on edges in $\bar{E}$, the backbone edges, have $r_{min} = \frac{3}{4}$. As all intersections (i.e. stations representing the nodes of $\bar{D}$) have at least one adjacent edge from the backbone, all those stations also have $r_{min} = \frac{3}{4}$.

Observe that the undirected communication graph of the minimal configuration, $G_{r_{min}}$, already has quite large connected components (cf. fig. 3.6): There

is one connected component corresponding to each edge in E', and the backbone is one connected component in the minimal configuration. For notational convenience, we refer to a component corresponding to an edge $e \in E'$ as an *edge-component*, or the *e-component*. We also use the terms 'incident' or 'adjacent' with edge-components or vertex stations when we mean that the corresponding edges or vertices have this property.

Let $M = cost(r_{min})$ be the cost of the minimal configuration, and let k be the number of vertices in a minimal vertex cover for D. We claim that a minimal range assignment with property CONNECTIVITY has cost $M + \gamma(m + k)$, where $\gamma = 1 - \left(\frac{3}{4}\right)^{\alpha}$, which implies the NP-hardness of CONNECTIVITY.

To prove our claim, we argue that, without loss of generality, in a minimal range assignment, only radii of node stations are increased, and those that are increased are increased by $\frac{1}{4}$, resulting in extra energy consumption γ.

First, we want to rule out that non-adjacent edge-components are directly connected to the same station. Assume conversely that l non-adjacent edge-components were connected via the same station. This means there would be l stations[1] S' with radii increased from $r_{min} \leq 1$ to at least to some value $C > 1$, costing at least $l(C^{\alpha} - 1)$. Instead, we could have connected those l edges to the backbone by increasing at most l radii to from $\frac{3}{4}$ to 1. In order for the first increase being cheaper,

$$l(C^{\alpha} - 1) \leq l\left(1 - \left(\frac{3}{4}\right)^{\alpha}\right) \iff C^{\alpha} + \left(\frac{3}{4}\right)^{\alpha} \leq 2$$

would have to hold. As $x + 1/x \geq 2 \ \forall x > 0$, this would imply that $C \leq \frac{4}{3}$, which means that some radii are increased to not more than this quantity. But by construction, it is not possible to connect non-adjacent edges with such a low radius, a contradiction to the assumption that all l edges are directly connected.

When an edge-component (or several adjacent edge-components) is connected to some node in the backbone, the cheapest way to achieve this is obviously to increase the radius of the station representing a node incident to it from $\frac{3}{4}$ to 1 (see also fig. 3.8). This is always cheaper than connecting adjacent edge-components directly:

Assume that two nodes on non-backbone edges would have increased radii in order be directly connected. The cheapest case of this would be if two adjacent such edges would have their last node's radius increased from 1 to $\sqrt{2}$ (see figure 3.7). To rule out this case (and in consequence, all other more expensive cases),

[1] possibly even $l + 1$, if they are connected via a backbone station

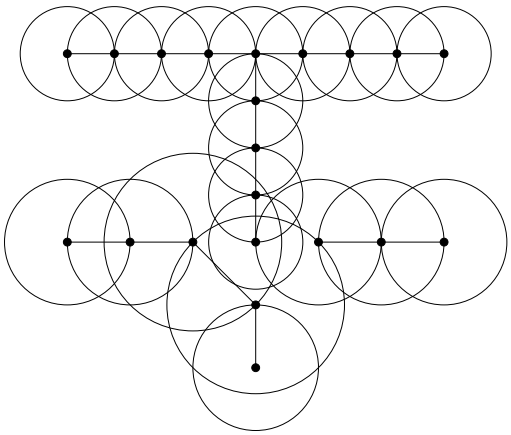

Figure 3.7: An unwanted connection scenario.

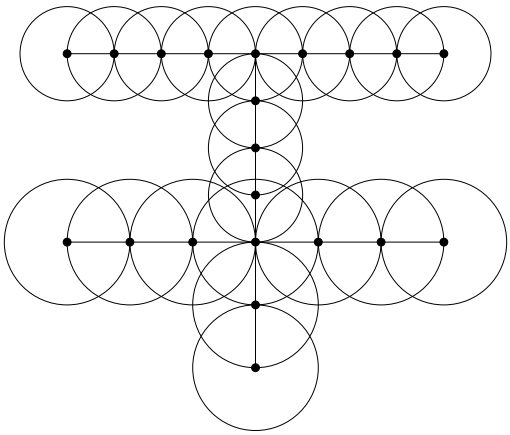

Figure 3.8: The cheapest way to connect edges. Automatically, all incident edges are attached.

we have to show that

$$\begin{aligned} \delta = 1 - \left(\frac{3}{4}\right)^{\alpha} &\leq 2(\sqrt{2}^{\alpha} - 1) \\ \Leftrightarrow \quad 3 &\leq 2\sqrt{2}^{\alpha} + \left(\frac{3}{4}\right)^{\alpha} \end{aligned}$$

which is easy:

$$2\sqrt{2}^{\alpha} + \left(\frac{3}{4}\right)^{\alpha} > 2\sqrt{2}^{\alpha} + \frac{1}{\sqrt{2}^{\alpha}} > \sqrt{2}^{\alpha} + 2 > 3.$$

We have now argued that in a minimal solution to CONNECTIVITY, only radii of stations corresponding to nodes of D' are increased from the minimal configuration, and if so, they are increased from $\frac{3}{4}$ to 1. Note that for nodes where this is the case, all incident edges are thereby connected to the backbone and its connected component (cf. fig. 3.8). So in order for the range assignment to be connected, these nodes have to form a vertex cover of D'. On the other hand, by virtue of the backbone, increasing each node in a vertex cover for D' already makes a minimal configuration connected.

Finally, we notice that by Lemma 3.3, a minimal vertex cover for D' is of size $m + k$, which means that a minimal range assignment with CONNECTIVITY has cost $M + \gamma(m + k)$, which concludes this proof.

□

3.3.2 NP-hardness for well-spread instances

Our reduction produces CONNECTIVITY instances where all stations lie on lines of a graph, not really spread-out in the available 2-d space, with many "holes" with no stations at all. It is nevertheless possible to make this construction well-spread, with the following simple trick.

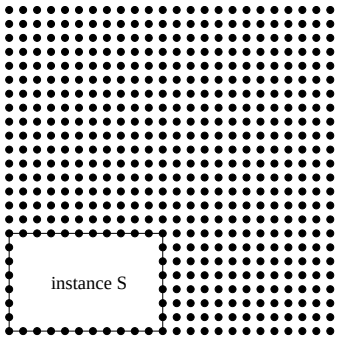

Figure 3.9: Making S well-spread by adding a well-spread grid.

The set of stations $S \subseteq \mathbb{R}^2$ is contained in its 'bounding box' R, the smallest axis-parallel rectangle such that all stations in S lie inside R. Let a be the longer side-length of R, and put a square Q of side-length $2a$ around R. S is completely enclosed by the backbone (cf. Fig. 3.4), i.e., in particular, stations lying on a regular grid with mesh distance $\frac{3}{4}$. When we fill up the rest of Q with such a grid, we get a well-spread instance. This is easy to see: Even if the lower left quadrant of Q would be completely empty, we get the following estimations. Scale the instance such that $\delta(S) = 1$, and assume Q has side-length l. Then $|S| = l^2$ and $\Delta(S) = l\sqrt{2}$, so choosing

$$c \leq \frac{3}{4} \cdot \frac{\delta(S)\sqrt{|S|}}{\Delta(S)} = \frac{3}{4}\frac{l}{l\sqrt{2}} = \frac{3}{4\sqrt{2}}$$

shows that we get well-spread instances. Obviously, in a minimal configuration the additional vertices all have radius $\frac{3}{4}$ and lie all in the same connected component as the backbone, and it does not make sense to increase any of their radii. It is obvious how to fill up such a well-spread square to a well-spread cube, proving

Theorem 3.6. *For any $\alpha > 0$,* CONNECTIVITY *on well-spread instances in $\mathbb{R}^2$ and $\mathbb{R}^3$ is NP-hard.*

□

3.3.3 APX-hardness of CONNECTIVITY in 3-d

The construction in the NP-hardness proof for CONNECTIVITY is far from being approximation-preserving: The fixed cost M of the minimal configuration is much

larger than the variable cost of the vertex cover. More precisely, M would have to be bounded by some (preferably low) constant factor times the number of vertices n. To the best of our knowledge, no orthogonal 3-d graph drawing method is known that uses only $O(n)$ total length, so we cannot hope to achieve this goal with this construction when $\alpha \leq 1$. However, the situation changes when $\alpha > 1$: Because the power function is now strictly convex, smaller radii cost far less than big radii. This means that we can make the power of the internal radii on the edges negligible again with the help of Lemma 2.25 by inserting a large number of stations on every edge at a very small distance.

As in the APX-hardness proof in the preceeding chapter, we again start our reduction from a low degree Vertex Cover problem and use the results of Chlebík and Chlebíková in Lemma 2.14.

Given a low-degree instance of Vertex Cover $D = (V, E)$, we describe how to build the graph $\bar{D}$ which later gets drawn in the Euclidean space $\mathbb{R}^3$. Note that we cannot use our reduction to prove APX-hardness of Range Assignment Problems in 2-d, because Planar Vertex Cover is not APX-hard, but there exists a PTAS for this problem [Bak94]. This time, as we do not have to observe planarity, the construction of the backbone becomes very simple: Let the vertices of V be given in some arbitrary order $V = \{v_1, \ldots, v_n\}$. The backbone vertices B contain one copy of each original vertex, say $B = \{v'_1, \ldots, v'_n\}$, and the backbone edges consist of one edge between each original node and its copy, and a cycle through all backbone nodes: $\bar{E} = \{\{v_i, v'_i\} \mid 1 \leq i \leq n\} \cup \{\{v'_i, v'_{i+1}\} \mid 1 \leq i \leq n-1\} \cup \{\{v'_n, v'_1\}\}$. Call $\bar{D} = (V \cup B, E \cup \bar{E})$. Given a constant $0 < \varepsilon < 1$, choose s according to Lemma 2.25 and construct a polynomial set of stations S in the following way:

- Construct an orthogonal drawing of $\bar{D}$ in $\mathbb{R}^3$.
- Scale the drawing by factor 3.
- For all polylines representing original edges $e \in E$, remove the first and last open unit interval of the polyline (i.e. do not erase any integer points).
- Replace all remaining unit line segments with $s + 1$ stations along this line at distance $1/s$.

Here, the scaling step is needed to ensure that at least one length unit of each edge remains. Note that when the maximum degree in D is Δ, the maximum degree of $\bar{D}$ will be $\Delta + 1$. So according to Lemma 3.4, we must have $\Delta \leq 5$. We use this set of stations in order to prove

Theorem 3.7. *For any $\alpha > 1$, it is NP-hard to approximate* Connectivity *in $\mathbb{R}^3$ within $1 + \frac{1}{260}$.*

Proof. Again, we first consider a minimal configuration r_{min} of S with cost M and its communication graph $G_{r_{min}}$. Due to the construction, all stations on backbone edges already lie in one connected component, and there is one component for each original edge. Similar arguments as in the proof for Theorem 3.5 show that in a minimal solution, all edges are directly connected to the backbone, and only vertex stations may have an increased radius of 1. In order to be connected, the nodes corresponding to these vertex stations must form a vertex cover for D. Additionally, in each edge component, one of the two end stations' radii must be increased to 1. On the other hand, we can always take one station which is close to a station from the node cover and whose radius thus is increased to 1, leading to a connected range assignment.

Note that because $\varepsilon < 1$, we have shown that there exists a node cover of size at most k for D iff there is a range assignment r for S costing no more than $k + m + \varepsilon$. In a graph with maximum degree Δ, the size of a vertex cover is at least m/Δ. We now assume that D has maximum degree 4, and due to the choice of s we have that

$$cost(r) \leq M + k + m \leq 5k + \varepsilon$$

So if we could approximate Connectivity in 3-d to $1 + \frac{1}{5 \cdot 52}$, this would allow us to construct a vertex cover of size at most

$$\left\lfloor k + \frac{1}{5 \cdot 52}(5k + \varepsilon) \right\rfloor = \left\lfloor (1 + \frac{1}{52})k + \frac{\varepsilon}{260} \right\rfloor \leq (1 + \frac{1}{52})k,$$

i.e., we could approximate 4–Vertex Cover up to $1 + \frac{1}{52}$. According to Lemma 2.14, this would solve an NP-hard problem. □

Note that we have chosen to reduce from 4–Vertex Cover, because this optimizes the trade-off between inapproximability result and reduction size.

3.3.4 APX-hardness for well-spread instances

This idea behind this proof is basically the same as in the NP-hardness proof for well-spread instances: We fill up the surrounding space with a grid of mesh distance δ. Note that in order to stay approximation-preserving, the additional stations must not be too expensive.

Theorem 3.8. *For any $\alpha > d \geq 3$, approximating* Connectivity *within $1 + \frac{1}{260}$ remains NP-hard even when restricted to well-spread instances.*

Proof. We deal here with the most interesting case, $d = 3$, but everything holds also for higher dimensions.

Let a be the largest of width, height and depth of the orthogonal drawing of the reduced Vertex Cover instance $\bar{D}$. In order to proof the claim, we show that it is possible to construct a cubical grid of stations C with side length $2a$ having

a connected range assignment costing no more than any constant ε in polynomial time.

We will place stations on this grid at distance $1/s$, so this cube will have $(2as+1)^3$ stations, and its minimal range assignment r_{min}, which is already connected, will have cost

$$cost(r_{min}) = \frac{(2as+1)^3}{s^\alpha}$$

Clearly, for constants $\alpha > 3$ and $\varepsilon < 1$ we can choose a polynomial s in such a way that $cost(r_{min}) < \varepsilon$. Note that this cubical grid is well-spread, even when we cut out a cube of sidelength $(a+1)$, and insert the set of stations constructed for $\bar{D}$ in the APX-hardness proof, with s as in the cube. Finally, we connect an arbitrary edge near to the grid stations with a line of stations at distance $1/s$ with the cut-out cube. Surely, the minimal configuration for this new set of stations will cost less than the one of the whole cube, so the equations from the above proof will still hold, implying the claimed result. □

This construction for APX-hardness of well-spread instances will work for all problems.

3.4 Hardness results for STRONG CONNECTIVITY

In the following two sections, we will adapt our reductions for CONNECTIVITY for the STRONG CONNECTIVITY and BROADCAST problems.

3.4.1 NP-hardness of STRONG CONNECTIVITY in 2-d

The reduction will be exactly the same as for CONNECTIVITY, but the proof will slightly differ. The main difference is that now directed links are established already when one of the two stations has a large enough radius.

Theorem 3.9. *For any $\alpha > 0$,* STRONG CONNECTIVITY *in $\mathbb{R}^2$ is NP-hard, already for well-spread instances.*

Proof. The minimal configuration r_{min} due to Lemma 1.9 is of course the same as in the proof of Theorem 3.5, but its directed communication graph $\vec{G}_{r_{min}}$ looks slightly different: The strongly connected components are the same as the connected components in $G_{r_{min}}$ (namely one for the whole backbone and one for each split edge), but additionally, each split edge component already has two outgoing arcs, one to each incident vertex station. (cf. fig. 3.10) Still, only vertex stations can have increased radii in a minimal solution: The cheapest alternative now increases one edge-end from 1 to $\sqrt{2}$ (see fig. 3.11), with additional cost

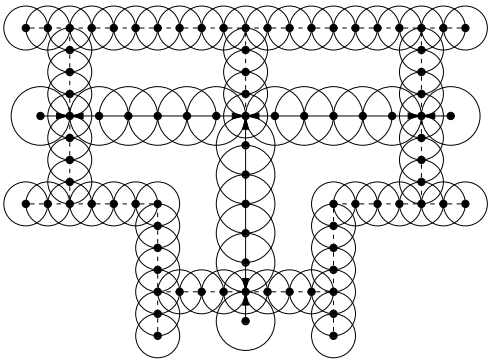

Figure 3.10: The directed communication graph of r_{min}.

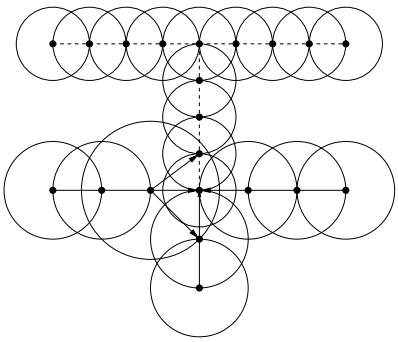

Figure 3.11: Now already one radius of $\sqrt{2}$ would suffice.

$\sqrt{2}^{\alpha} - 1$ instead of $1 - \left(\frac{3}{4}\right)^{\alpha}$, which, for all α, saves us nothing. And if $\alpha \geq 1$, it is obvious that no radius is increased to more than 1.

The situation changes here when $\alpha < 1$: As links do not have to be bidirectional, and now the cost function is strictly concave, it could pay off to increase just one radius to some large value $C \gg 1$ to send data to many (or possibly all) stations. To be on the safe side, we scale the instance again by a factor of $\sqrt[\alpha]{n}$, which will ensure that

$$C^{\alpha} - 1 \geq n\gamma$$

holds for radius $C \geq \frac{4}{3}\sqrt[\alpha]{n}$, meaning that it is already cheaper to increase every original vertex to 1 than one single radius to more than $\frac{4}{3}\sqrt[\alpha]{n}$, which would after scaling not reach any new component.
The proof for well-spread instances is the same as in Theorem 3.6. □

3.4.2 APX-hardness of Strong Connectivity in 3-d

The only difference to the construction for Connectivity is that now the outgoing arc of an edge does not have to be parallel to the ingoing arc. If some edge-end is increased to 1, and the incident vertex is not, it could indeed be cheaper to increase the border station further to $\sqrt{2}$ and send to another adjacent edge. (cf. fig. 3.12) This could indeed be cheaper if $\alpha < 2$, and as this is the only thing we have to worry about, let us assume in the following that $1 < \alpha < 2$.

We will now present a slightly changed reduction: To the original Vertex Cover instance $D = (V, E)$, no new vertices but only new edges will be added. For vertex set $V = \{v_1, \ldots, v_n\}$, add a directed Hamiltonian cycle $\bar{E} = \{(v_i, v_{i+1}) \mid 1 \leq i \leq n-1\} \cup \{(v_n, v_1)\}$ as the backbone, already completing the construction of $\bar{D} = (V, E \dot{\cup} \bar{E})$. The direction of the added arcs is needed only for notational convenience later on. Note that $\bar{D}$ may contain parallel backbone and original edges, which is not a problem.

The construction is now similar to the one before; the new thing is that we also erase some part of the backbone lines:

- Let $\beta = \sqrt[\alpha]{2 - \sqrt{2}^{\alpha}}$ and choose an $\varepsilon < 1 - \beta^{\alpha}$; choose an s according to Lemma 2.25.
- Construct a polynomial orthogonal drawing of $\bar{D}$ in $\mathbb{R}^3$.
- Scale the drawing by factor 3.
- For all polylines representing original edges $e \in E$, erase the first and last open unit interval of the polyline (i.e., do not erase any integer points).
- For all polylines representing backbone edges $e \in \bar{E}$, erase the first (i.e., the interval starting at the tail of e) open interval of length σ/s, where $\sigma \in \mathbb{N}$ is chosen such that $(\sigma - 1)/s < \beta \leq \sigma/s$. Again, erase neither the first nor the last point of this interval.
- Let $\varepsilon < 1 - \beta^{\alpha}$, and replace all remaining line segments of length 1 (resp. $1 - \sigma/s$) with $s + 1$ (resp. $s - \sigma + 1$) stations along this line at distance $1/s$.

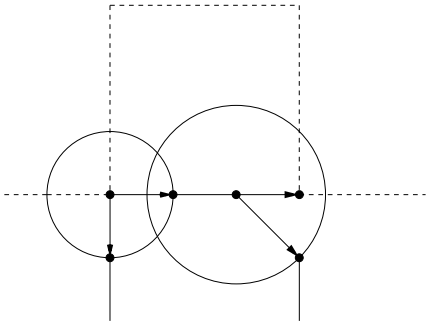

Figure 3.12: A cheaper way to connect edges when $\alpha < 2$.

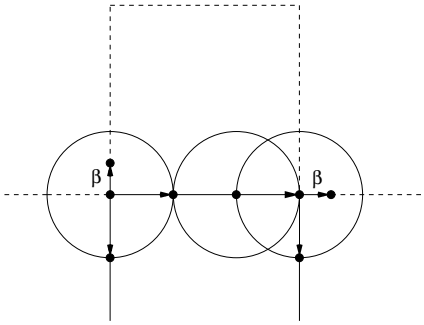

Figure 3.13: By leaving spaces of length β in the backbone, this problem is fixed.

So this time, the minimal configuration consists of one strongly connected component for each original or backbone edge. Note that $0 < \beta < 2 - \sqrt{2} < 0.6$. In order for S to be strongly connected, at least one of the end stations of each backbone edge must have radius at least β; setting all vertex radii to β suffices to make the backbone strongly connected. As it has no advantage not to do so, we assume this is the case.

This means now every vertex station wlog. already has radius β. If we would now choose to transmit data with an edge station, we would pay at least additional cost $\sqrt{2}^{\alpha} - 1$ instead of paying $1 - \beta^{\alpha}$ for an incident vertex station, which due to the choice of β is not cheaper (see also fig. 3.13). Hence, we can assume that

only vertex stations are used for connectivity, and the vertex stations with radius 1 in a minimal range assignment with STRONG CONNECTIVITY form a minimal vertex cover. This means a minimal range assignment r costs

$$cost(r) \leq n\beta^\alpha + k(1-\beta^\alpha) + m + \varepsilon$$

where k is the size of a minimal vertex cover for D. As the hardness results in Lemma 2.14 also hold for regular instances, we can assume that $k \geq m/\Delta = n/2$. We choose to reduce from 4–Vertex Cover, and normalize the costs for r s.t. the cost for each node in the vertex cover is 1:

$$k + \frac{n\beta^\alpha + m}{1-\beta^\alpha} \leq k + \left(\frac{2+\beta^\alpha}{1-\beta^\alpha}\right) 2k + \frac{\varepsilon}{1-\beta^\alpha} < \left(\frac{7-\sqrt{2}^\alpha}{\sqrt{2}^\alpha - 1}\right) k + 1$$

which finally concludes the proof of

Theorem 3.10. *It is NP-hard to approximate* STRONG CONNECTIVITY *in* $\mathbb{R}^3$ *within* $1 + \frac{1}{260}$*, if* $\alpha \geq 2$*, and within* $1 + \frac{\sqrt{2}^\alpha - 1}{(7-\sqrt{2}^\alpha)\cdot 52}$*, if* $1 < \alpha < 2$*.*
For any $\alpha > d \geq 3$*, approximating* CONNECTIVITY *within* $1 + \frac{1}{260}$ *remains NP-hard even when restricted to well-spread instances.*

3.5 Hardness results for BROADCAST

The main difference to the two preceding problems is that we cannot make use of Lemma 1.9, because no node (except for the source node s) has to increase its radius a priori. Indeed, the optimal solution when $0 < \alpha \leq 1$ is to have s directly broadcast to all stations, and all other stations have radius 0, so we assume $\alpha > 1$. This time, we use the APX-type construction also for proving the NP-hardness results.

Theorem 3.11. *For any* $\alpha > 1$*,* BROADCAST *in* $\mathbb{R}^2$ *is NP-hard, already for well-spread instances.*

Proof. Given an instance of planar 3–Vertex Cover D, construct $\bar{D}$ as in Theorem 3.5, draw this orthogonally in the plane, scale it by factor 3, remove the first and last unit of non-backbone edges, and replace the lines by an appropriate number of stations s.t. $\varepsilon < 1$. Let the source station s be any backbone station. In order to broadcast to all stations, we have to set the vertex stations of a vertex cover of D' to radius 1. This time, we cannot argue with the minimal configuration, but as every (STRONG) CONNECTIVITY solution is also feasible for BROADCAST, we can be sure that the overhead cost of the construction does not exceed ε. This means there will be a range assignment of size $< m + k + 1$ iff there is a vertex cover of D' of size $m + k$, showing NP-hardness of BROADCAST.

Again, this construction can be made well-spread with the same method as in Theorem 3.6, where the grid now has mesh-distance $1/s$. Note that despite of the now larger overhead, everything is still polynomial. □

Theorem 3.12. *For any $\alpha > 1$, approximating* Broadcast *in $\mathbb{R}^3$ better than $1+\frac{1}{50}$ is NP-hard. For $\alpha > 3$, the same result holds also for well-spread instances.*

Proof. We use exactly the same construction as in the proof for Theorem 3.7, and let the source station be some backbone station. As in the preceding proof, this time exactly the vertex stations of a vertex cover have to be increased to radius 1; no additional edge stations have to be significantly increased. This means a range assignment of size less than $k + \varepsilon$ leads to a vertex cover of size k, so we only have overhead ε, which we can make arbitrarily small. So this time it pays to reduce from 5-Vertex Cover, yielding the claimed result. □

3.6 Overview and conclusion

In Table 3.1, old and new results for the investigated range assignment problems and ranges of α and d are given. New results in this thesis are listed in bold print.

We prove NP-hardness results for (Strong) Connectivity and Broadcast for low values of α, in particular for (Strong) Connectivity for $\alpha \leq 1$, i.e. with $\triangle$-inequality. This and the first overall APX-hardness result for Broadcast answer open questions posed in the survey [CHP+02]. Thus, we could fill the remaining gaps concerning NP-hardness of all problems. Our simpler and more adjustable constructions yield improved inapproximability results for all APX-hard cases. We could also give the first hardness results for well-spread instances.

What remains open is the approximability status for these problems in 2-d and for $\alpha \leq 1$, i.e., the question whether these problems are also APX-hard for $d = 2$ and/or $\alpha \leq 1$, or if there is/are some cases which allow a PTAS. The problem with 2-d is that we need an APX-hard *planar* problem to reduce from, and for $\alpha \leq 1$, we cannot use Lemma 2.25 to reduce the reduction overhead to make it approximation preserving. The latter issue seems to be serious indeed, and leads us to conjecture that these problems should have a PTAS for Euclidean distances. From the previous chapter, we know that it is even still unclear whether non-geometric Strong Connectivity allows a PTAS. But the ideas of Arora [Aro98] and Mitchell [Mit99], the now 'standard' geometric PTASs for various geometric problems, seem not to be applicable. This is due to the new cost function: we do not simply add edge costs, but only the *longest* edge incident to a node. This makes the cost function not only non-linear, but also *'non-local'*. By this we mean that a slight change in some location of the instance may change influence the situation somewhere completely else. If there was a PTAS, it is

<table>
<tr><th></th><th>$d = 2$ (old)</th><th>$d \geq 3$ (old)</th><th>$d = 2$ (new)</th><th>$d \geq 3$ (new)</th></tr>
<tr><td>C, $0 < \alpha \leq 1$</td><td>—</td><td>—</td><td>NP-hard
(also for w.s.i.)</td><td>NP-hard
(also for w.s.i.)</td></tr>
<tr><td>C, $1 < \alpha < 2$</td><td rowspan="2">— *</td><td rowspan="2">— *</td><td rowspan="2">NP-hard
(also for w.s.i.)</td><td>APX-hard
($\boldsymbol{\rho = 1 + \frac{1}{260}}$)</td></tr>
<tr><td>C, $\alpha > d$</td><td>(also for w.s.i.)</td></tr>
<tr><td>SC, $0 < \alpha < 1$</td><td>—</td><td>—</td><td rowspan="4">NP-hard
(also for w.s.i.)</td><td>NP-hard
(also for w.s.i.)</td></tr>
<tr><td>SC, $\alpha = 1$</td><td>—</td><td rowspan="2">NP-hard
[KKKP00]</td><td>NP-hard
(also for w.s.i.)</td></tr>
<tr><td>SC, $1 < \alpha < 2$</td><td>—</td><td>APX-hard
($\boldsymbol{\rho = 1 + \frac{\sqrt{2}^{\alpha}-1}{(7-\sqrt{2}^{\alpha})\cdot 52}}$)</td></tr>
<tr><td>SC, $\alpha \geq 2$</td><td rowspan="2">NP-hard
[CPS04]</td><td rowspan="2">APX-hard
[CPS04]
($\rho = 1 + \frac{1}{495}$)</td><td>NP-hard
(also for w.s.i.)</td><td>APX-hard
($\boldsymbol{\rho = 1 + \frac{1}{260}}$)</td></tr>
<tr><td>SC, $\alpha > d$</td><td>(also for w.s.i.)</td></tr>
<tr><td>B, $1 < \alpha < 2$</td><td>—</td><td>—</td><td>NP-hard
(also for w.s.i.)</td><td rowspan="2">APX-hard
($\boldsymbol{\rho = 1 + \frac{1}{50}}$)</td></tr>
<tr><td>B, $\alpha \geq 2$</td><td rowspan="2">NP-hard
[CCP+01b]</td><td rowspan="2">NP-hard
[CCP+01b]</td><td rowspan="2">NP-hard
(also for w.s.i.)</td></tr>
<tr><td>B, $\alpha > d$</td><td>(also for w.s.i.)</td></tr>
</table>

(* NP-hardness for $\alpha \geq 2$ is implicit in [CPS04].)

Table 3.1: List of previous and new results for different Range Assignment Problems

our suspicion that it should result from a completely new approach. It would be worthwile to investigate this direction, as it may produce a new line of PTASs for other range assignment problems.

Chapter 4

Approximation algorithms

4.1 The MST heuristic

4.1.1 (Strong) Connectivity

As we already mentioned when introducing range assignment problems, you can view these problems as traditional network design problems with a new cost function. We used the suggestive phrase of a "change of meric". It is already interesting by itself how much difference there can be between an optimal range assignment and a traditional MST. It turns out to be a very good approximation already for (Strong) Connectivity, due to the lower bound proved in Chapter 1. This fact has been observed already in [KKKP00].

Theorem 4.1 ([KKKP00]). *An MST is a 2-approximation for* (Strong) Connectivity.

Proof. Basically, this holds because each edge dominates at most two nodes, and $|MST|$ is a lower bound for an optimal range assignment.

Let $OPT = OPT(G)$ be an optimal range assignment for instance G, and $MST = MST(G)$ an MST for G. Recall that for Strong Connectivity, we identify an edge $\{v, w\}$ with the pair of arcs (v, w) and (w, v). We have

$$\begin{aligned} cost(MST) &= \sum_{v \in V} \max_{(v,w) \in MST} d(v, w) \\ &\leq \sum_{v \in V} \sum_{(v,w) \in MST} d(v, w) = 2 \cdot |MST| \\ &\leq 2 \cdot cost(OPT). \end{aligned}$$

The latter inequality is the lower bound in Lemma 1.11. Note that the first inequality uses the fact that we always assume non-negativity. □

So, an MST is not only a 2-approximation for CONNECTIVITY, but also STRONG CONNECTIVITY. We just want to remark that the strongly connected analogon of an MST, a minimum strongly connected spanning subgraph, is also a 2-approximation for STRONG CONNECTIVITY.

Corollary 4.2. *A minimum strongly connected spanning subgraph is a 2-approximation for* STRONG CONNECTIVITY.

Proof. Let $\vec{M}$ be a minimum strongly connected spanning subgraph of some instance. Each arc dominates at most one node, i.e. is counted at most once in $cost(\vec{M})$. Taking both directions of each edge in MST yields a strongly connected spanning subgraph $\overrightarrow{MST}$, which is of course at least as expensive as $\vec{M}$ regarding the usual $\sum$ cost function. So we have

$$\begin{aligned} cost(\vec{M}) &= \sum_{v \in V} \max_{(v,w) \in \vec{M}} d(v,w) \\ &\leq \sum_{v \in V} \sum_{(v,w) \in \vec{M}} d(v,w) \leq \sum_{v \in V} \sum_{(v,w) \in \overrightarrow{MST}} d(v,w) \\ &= 2 \cdot |MST| \leq 2 \cdot cost(OPT). \end{aligned}$$

□

Of course as the MINIMUM STRONGLY CONNECTED SPANNING SUBGRAPH PROBLEM is NP-hard, this Corollary does not give rise to an efficient approximation algorithm. In fact, even if we had fast access to such an MSCSS, this would not give us a worst case for STRONG CONNECTIVITY than the MST-heuristic, already in a very restricted setting.

Let us stress again that an MST is a 2-approximation for (STRONG) CONNECTIVITY for *any* non-negative symmetric distance function. However, we will see now that this factor is tight already in a very simple setting, namely the real line with Euclidean distances. This has also been observed in [ACM+03].

Theorem 4.3. *Approximation factor 2 for the MST-heuristic for* (STRONG) CONNECTIVITY *is asymptotically tight already on the real line with Euclidean distances.*

Proof. We construct a family of instances where $cost(MST)$ approaches $2 \cdot OPT$, depending on parameters $\epsilon > 0$ and $k \in N$. Instance $I_{k,\varepsilon}$ looks as follows:

$$I_{k,\varepsilon} = \{\varepsilon, 1, 1+\varepsilon, 2, 2+\varepsilon, \ldots, k-1, k-1+\varepsilon, k\}$$

These instance basically consists of pairs of very close points at distance 1 to another, except for single points at the left and right end of the instance. Obviously,

the MST of points on the line consists of all intervals. Each node has a length $(1-\varepsilon)$ edge incident to it, so

$$cost(MST(I_{k,\varepsilon})) = 2k(1-\varepsilon) = 2k - 2k\varepsilon \approx 2k$$

In an optimal range assignment, one of the nodes in each pair transmits to both neighboring pairs with radius 1, while the other is just connected to its ε-partner at radius ε.

Figure 4.1: An instance $I_{k,\varepsilon}$ with its MST (straight lines) and OPT (dashed lines).

An optimal range assignment OPT can be seen in Figure 4.1. It has cost

$$cost(OPT(I_{k,\varepsilon})) = k + (1-\varepsilon) + (k-1)\varepsilon = k + 1 + (k-2)\varepsilon \approx k + 1$$

implying that

$$\lim_{\substack{k\to\infty \\ \varepsilon\to 0}} \frac{cost(MST(I_{k,\varepsilon}))}{cost(OPT(I_{k,\varepsilon}))} = 2$$

□

This holds for the MSCSS-heuristic for Strong Connectivity as well, as MSCSS and MST are identical on these instances.

So, the simple analysis for the MST-heuristic is already tight in a very restricted setting. However, it does not yet fully use the lower bound of Lemma 1.11. When we examine the MST approximation ratio depending on n, the number of stations, it directly allows the following calculation, where $w = \max_{e\in MST} w(e)$ is the largest weight in an MST:

$$\frac{cost(MST)}{cost(OPT)} \leq \frac{2 \cdot mst}{mst + w} = \frac{2(mst + w)}{mst + w} - \frac{2w}{mst + w} = 2 - \frac{2w}{nw} = 2 - \frac{2}{n}$$

By the way, this term is quite common for the approximation ratio of MST-heuristics for other problems. Just to mention two, MST-heuristics for Metric TSP (e.g. [RSL77]) and the Steiner Tree Problem (e.g. [PS02]) have this performance guarantee, and it is tight in these two cases.

Interestingly, it is not tight here. With a more careful calculation than the one above, we get the following tight bound on the approximation ratio of the MST-heuristic in dependence of n.

Theorem 4.4. *An MST is a feasible solution for* (Strong) Connectivity *which is at most a factor*

$$2 - \frac{2}{\lfloor \frac{n}{2} \rfloor + 1}$$

more expensive than an optimal solution. This bound is tight in dependence of n*.*

Proof. We take a closer look on how edges determine the range of vertices. An edge can dominate two, one or no vertex. Let A be the set of edges in an MST that dominate two vertices, B the set that dominate exactly one vertex, and C the set that dominate no vertex at all. Again, w is the weight of a largest edge in MST.

For the approximation ratio of the MST-heuristic, the following estimation holds:

$$\begin{aligned} \frac{cost(MST)}{mst + w} &= \frac{2|A| + |B|}{|A| + |B| + |C| + w} \\ &\leq \frac{2|A|}{|A| + |C| + w} \leq \frac{2|A|}{|A| + w} \\ &= 2 - \frac{2w}{|A| + w} = 2 - \frac{2}{\frac{|A|}{w} + 1} \end{aligned}$$

For the term $\frac{|A|}{w}$, we have:

$$\frac{|A|}{w} = \sum_{a \in A} \frac{w(a)}{w} \leq \sum_{a \in A} 1 = \#A \leq \left\lfloor \frac{n}{2} \right\rfloor,$$

proving the approximation ratio.

To see that this ratio is tight, we consider instances $I_{k,\varepsilon}$ again. Note that they all have an even number of nodes $n = 2k$. For $\varepsilon \to 0$, we have:

$$\frac{cost(MST(I_{k,\varepsilon}))}{cost(OPT(I_{k,\varepsilon}))} \approx \frac{2k}{k+1} = \frac{n}{\frac{n}{2}+1} = \frac{n+2}{\frac{n}{2}+1} - \frac{2}{\frac{n}{2}+1} = 2 - \frac{2}{\frac{n}{2}+1}$$

For odd $n = 2k+1$, consider instance $I_{k,\varepsilon} \cup \{k+\varepsilon\}$. For $\varepsilon \to 0$, this instance has about the same cost for MST and OPT as $I_{k,\varepsilon}$, namely $cost(MST) = 2k = n-1$ and $cost(OPT) = k+1 = (n+1)/2$. In this case, we calculate:

$$\frac{cost(MST(I_{k,\varepsilon}))}{cost(OPT(I_{k,\varepsilon}))} \approx \frac{2k}{k+1} = \frac{n-1}{\frac{n}{2}+\frac{1}{2}} = \frac{n+1-2}{\frac{n}{2}+\frac{1}{2}} = 2 - \frac{2}{\frac{n}{2}+\frac{1}{2}} = 2 - \frac{2}{\lfloor \frac{n}{2} \rfloor + 1}$$

showing tightness of the bound for all n. □

4.1.2 Broadcast

For the sake of completeness, we mention that an MST-heuristic yields the best known algorithm for Geometric Broadcast. The Broadcast MST-heuristic takes an MST and "hangs it up" by source node s, i.e. directs edges in MST away from s. The MST heuristic yields a constant factor approximation for constant dimension d and $\alpha \geq d$.[CCP+01a] However, this factor is expontial in d.

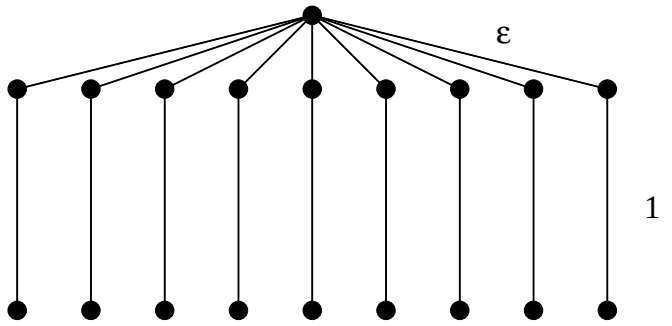

Figure 4.2: A Broadcast instance G_k which is bad for the MST-heuristic.

At first, observe that this heuristic performs very badly in general graphs. Consider the following family of graphs $G_k = (V, E, w)$ with $V = \{s, i_1, \ldots, i_k, o_1, \ldots, o_k\}$ and $E = \{\{s, i_j\} \mid j = 1, \ldots, k\} \cup \{\{i_j, o_j\} \mid j = 1, \ldots, k\}$. Weights w are $d(s, i_j) = \varepsilon$ for all j and $d(i_j, o_j) = 1$. Thus, G_k looks like a star with k outer vertices o_j but also has k inner vertices i_j. See Figure 4.2. Now let d be the metric induced by G_k. The MST heuristic applied to d assigns the k inner vertices radius 1 at total cost k, while it is optimal to set $r(s) = 1+\varepsilon$ and $r(v) = 0$ for $v \neq s$. Thus, the MST heuristic performs like $n/2$ on these instances.

Now to geometric cases. In case $\alpha < d$, it is easy to see that this heuristic does not have a constant factor approximation: Take a d-dimensional cubical grid with sidelengths k, and the central node as source node s. The MST solution (every node has radius 1) has cost k^d, while sending from the center everywhere directly costs $(k/\sqrt{2})^\alpha = O(k^\alpha)$. Thus, for $\alpha < d$, the ratio of these two expressions cannot be bounded by a constant.

In case $d = 2$, a series of papers was devoted to improving the approximation factor of the MST heuristic. Finally, factor 6 was shown by Ambühl [Amb05], closing the gap to a lower bound for this heuristic. We briefly describe this lower bound here; see Figure 4.3 for the lower bound of 6 in $\mathbb{R}^2$. In general, the *kissing number* is the largest number of unit spheres that can be arranged around one unit sphere while still touching ("kissing") it. Basically, this kissing number tells us how much space we have to build instances like G_k above. In $\mathbb{R}^2$, we can build G_6 which you can see in Figure 4.3 which shows that Ambühl's analysis of the MST-heuristic is tight. Generally, in R^d the kissing number grows exponential

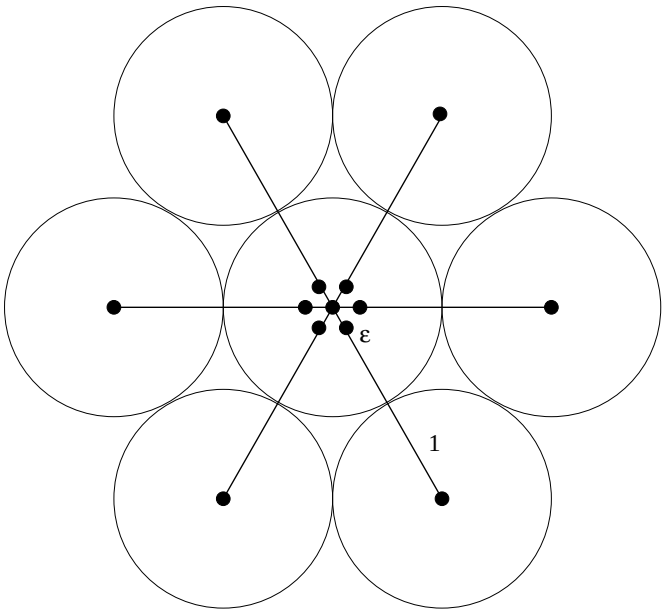

Figure 4.3: 6 circles in the plane "kissing" a unit circle, and inducing the worst-case instance for the MST heuristic.

with dimension d and implies a lower bound on the performance of MST which is exponential in d.

Future research work is the design of better approximation algorithms, or in fact any constant factor approximations for cases $1 < \alpha < d$. To start with, such an algorithm for BROADCAST in $\mathbb{R}^2$ for $1 < \alpha < 2$ would be interesting.

4.2 Greedy heuristics

Although an MST can be obtained via a greedy algorithm, and we are looking for minimum cost spanning trees in CONNECTIVITY, the MST heuristic is *not* a greedy algorithm for CONNECTIVITY, or any other of the range assignment problems we are addressing here. This is because an MST is the solution of a greedy strategy w.r.t. the usual cost function $|\cdot|$.

We consider two greedy strategies for the CONNECTIVITY problem, one corresponding to Prim's and another corresponding to Kruskal's greedy algorithm for computing an MST. Both algorithms are natural greedy strategies for the CONNECTIVITY problem. As a first observation, we will see that they are near-optimal on the worst-case example for the MST heuristic. One would hope that they should perform at least as well as the MST heuristic, and perhaps even better. We show that both algorithms have approximation ratio 2, and give instances showing this bound is tight. However, it is trickier to fool these algo-

rithms, as these instances are not quite as simple as the tight instances for the MST heuristic, and need more complicated settings.

4.2.1 Greedy like Kruskal

Figure 4.4 describes the range assignment analog of Kruskal's greedy strategy for computing minimum spanning trees in graphs.[Kru56] It iteratively adds the "cheapest" edge inducing no cycle until a spanning tree is constructed. Here, the "cheapest" edge is the one whose addition causes the least increase in cost of the currently induced range assignment. Adding large edges to a node which already has a high radius becomes cheaper than adding a large edge to a node with a low radius. Note that this means that the cost of adding the same edge to the current solution may change after each step of the algorithm, in contrast to Kruskal's (or Prim's) MST algorithm, where one basically goes through a static list where the edges are sorted by their weights.

Algorithm KRUSKAL

Instance: A weighted graph $G = (V, E, d)$ with distances $d : E \to \mathbb{R}_+$.

Output: A spanning tree $K \subseteq E$ of G.

Algorithm:

- **Let** $i := 0$, $K_i := \varnothing$.
- **Repeat**
  - **Let** $i := i + 1$
  - **Choose** $k_i \in E$ s.t. $K_{i-1} \cup \{k_i\}$ contains no cycle and k_i minimizes

$$\Delta(k) := cost(K_{i-1} \cup \{k\}) - cost(K_{i-1}).$$

  - **Let** $K_i := K_{i-1} \cup \{k_i\}$.
- **Until** $i = n - 1$.
- **Output** $K := K_{n-1}$.

Figure 4.4: Algorithm KRUSKAL.

To get an idea how this algorithm works, consider the worst-case instances $I_{k,\varepsilon}$ (cf. Figure 4.1) for the MST heuristic. First, all ε edges are added, what Kruskal's MST-algorithm would do as well. Then, a first length $1 - \varepsilon$ edge is chosen, say $\{\varepsilon, 1\}$. What happens now is that due to the increased range in node 1, adding edge $\{1, 2\}$ costs only about 1, while adding an MST edge, e.g. $\{1 + \varepsilon, 2\}$, costs

about 2. Thus, edge $\{1,2\}$ is selected, making edge $\{2,3\}$ much cheaper than $\{2+\varepsilon\}$, and so on. This behavior leads to a near-optimal solution (the choice of the first edge may be non-optimal, at negligible extra cost).

We now show that this algorithm has approximation ratio at most 2, as the MST heuristic. Indeed, the proof uses the lower bound $|MST|$ as well.

Theorem 4.5. *For the solution $K = K_{n-1}$ of greedy algorithm* KRUSKAL, *$cost(K) \leq 2 \cdot |MST|$ holds, implying that K is a 2-approximation for* STRONG CONNECTIVITY.

Proof. Let $MST = \{e_1, \ldots, e_{n-1}\}$ be a minimum spanning tree in G with edges sorted in increasing length, i.e., $w(e_1) \leq w(e_2) \leq \ldots \leq w(e_{n-1})$. We prove the theorem by induction. The induction hypothesis is:

$$\rho(K_i) \;\leq\; 2\sum_{j=1}^{i} w(e_j).$$

In the first step, the increase in power cost of each edge amounts to twice its length, so we select a cheapest edge. This means $w(k_1) = w(e_1)$, implying $cost(\{k_1\}) = w(e_1)$, thus the induction hypothesis holds at the beginning.

Assume now the hypothesis holds after step i. We choose k_{i+1} minimizing $\Delta(k)$. We know that K_i is a forest in G, and E_{i+1} is a larger forest. Thus we know (using the fact that the forests in G form a matroid) that there is at least one edge $\hat{e} \in E_{i+1}$ such that $K_i \cup \{\hat{e}\}$ is a forest. This means the greedy algorithm has considered using $\hat{e}$. Thus we have:

$$\Delta(k_{i+1}) \leq \Delta(\hat{e}) \leq \max_{e \in E_{i+1}} \Delta(e) \leq \max_{e \in E_{i+1}} 2w(e) = 2w(e_{i+1}).$$

Combining this with the induction hypothesis yields

$$cost(K_{i+1}) = cost(K_i) + \Delta(k_{i+1}) \leq 2\sum_{j=1}^{i} w(e_j) + 2w(e_{i+1}) = 2\sum_{j=1}^{i+1} w(e_j),$$

and we are done. □

4.2.2 Greedy like Prim

The PRIM greedy algorithm is started with a trivial component, namely an arbitrary vertex $v_0 \in V$, and grows this component successively until it contains V completely. Of course, always the (currently) cheapest edge is selected for growing this component; see Fig. 4.5.

Consider this algorithm again on the MST worst-case instances $I_{k,\varepsilon}$. Assume it is started at $v_0 = \varepsilon$. At first, edges $\{\varepsilon, 1\}$ and $\{1, 1+\varepsilon\}$ are selected, and Prim's

Algorithm PRIM

Instance: A weighted graph $G = (V, E, d)$ with distances $d : E \to \mathbb{R}_+$.

Output: A spanning tree $P \subseteq E$ of G.

Algorithm:

- **Let** $i := 0$, $V_0 := \{v_0\}$ an arbitrary start vertex $v_0 \in V$, $P_i := \varnothing$.
- **Repeat**
  - **Let** $i := i + 1$
  - **Choose** $p_i = \{v, w\} \in E$ s.t. $v \in V_{i-1}$, $w \notin V_{i-1}$, and p_i minimizes

$$\Delta(p) := cost(P_{i-1} \cup \{p\}) - cost(P_{i-1}).$$

  - **Let** $P_i := P_{i-1} \cup \{p_i\}$, $V_i := V_{i-1} \cup \{w\}$.
- **Until** $i = n - 1$.
- **Output** $P := P_{n-1}$.

Figure 4.5: Algorithm PRIM.

MST algorithm would now choose $\{1 + \varepsilon, 2\}$. Heuristic PRIM selects edge $\{1, 2\}$ instead, as node 1 has increased radius already, making this edge cheaper. In this way, PRIM finds a near-optimal solution for instances $I_{k,\varepsilon}$.

Algorithm PRIM also achieves an approximation ratio of 2. The proof is a bit more technical than the one for KRUSKAL.

Theorem 4.6. *For the solution P of the Prim greedy algorithm, $cost(P) \leq 2 \cdot mst$ holds, implying P is a 2-approximation for* (STRONG) CONNECTIVITY.

Proof. Again, we compare our solution with a minimum spanning tree MST. In addition to the P_i and V_i, we keep track of two more edge sets. Namely $E_i \subset MST$, where $E_i = \{e_1, \ldots, e_i\}$ contains the edges of M which have been *accounted* for. This notion will become clear later. Let $\overline{E_i} = MST \setminus E_i$ be the edges of MST which have not been accounted for. Finally, let $T_i = P_i \cup \overline{E_i}$.
This theorem is again proven by induction. The induction hypothesis is:

$$cost(P_i) \leq 2 \sum_{j=1}^{i} w(e_j), \quad \text{and } T_i \text{ is a spanning tree of } G.$$

In step one, the algorithm chooses a shortest edge incident to v_0 as p_1. If this edge is unique, it is clear that $e_1 = p_1$ is also contained in MST, and $cost(p_1) = 2w(e_1)$,

and $T_1 = (MST \setminus \{e_1\}) \cup \{p_1\} = MST$. Otherwise, adding p_1 to MST will close a (non-trivial) cycle. Choose the edge contained in MST on this cycle which is incident to v_0 as e_1. Obviously, $w(e_1) = w(p_1)$, or MST would not be minimal. Observing that $T_1 = (MST \setminus \{e_1\}) \cup \{p_1\}$ is again a (minimum) spanning tree, the induction hypothesis holds in the beginning.

Let the hypothesis hold after step i, and the Prim Greedy has chosen p_{i+1}. In case $p_{i+1} \in \overline{E_i}$, we let $e_{i+1} = p_{i+1}$ and are already done. So we assume that $\{v, w\} = p_{i+1} \notin \overline{E_i}$. This means that $T_i \cup \{p_{i+1}\}$ contains exactly one cycle C, consisting of p_{i+1} and the (v, w)-path in T_i. This path runs from V_i to $V \setminus V_i$, so it has to cross this cut at least once with some edge. By definition, this edge has to lie in $\overline{E_i}$, so we may choose it as e_{i+1}. This edge was considered by the Greedy algorithm but it rather chose p_{i+1}, so we have:

$$\Delta(p_{i+1}) \leq \Delta(e_{i+1}) \leq 2w(e_{i+1}).$$

(Observe that now e_{i+1} has been accounted for, explaining the terminology at the beginning.)

Concluding, as $T_{i+1} = (T_i \setminus \{e_{i+1}\}) \cup \{p_{i+1}\}$ remains a spanning tree, the induction hypothesis is proven, implying the theorem for $P = P_{n-1}$. □

4.2.3 Factor 2 is tight

One could think that, as both greedy strategies are not fooled by the worst-case instances for the MST heuristic, and the proofs for approximation ratio 2 rely on lower bound $|MST|$, that factor 2 is not the best we can show for these algorithms. However, we will construct instances where both algorithms asymptotically do not perform better than 2.

A linear example

First, we describe instance family consisting of points lying on the real line $\mathbb{R}$. However, distances are not Euclidean, nor according to Geometric Range Assignment Problems, i.e. Euclidean distances to the power α, so it is not immediate what it means that these points are lying on a line. To make this clearer, we define the concept of *monotone* (or *linear*) distances.

Definition 4.7. A Range Assignment instance $G = (V, V \times V, d)$ is called *monotone* or *linear*, if there is some total order $\prec$ on the vertex set V satisfying the following condition.
For any three points $v \prec w \prec u$, we have:

$$d(v, w) \leq d(v, u) \quad \text{and} \quad d(w, u) \leq d(v, u)$$

This definition means that points are linearly arranged such that if we reach points l_v and r_v on the left and right from node v, we reach every node between l_v and r_v from v. Another way to express this: The directed adjacency matrix of any range assignment, indexed in order $\prec$, where row v represents the outgoing arcs from v, has rows with the *consecutive ones* property.

Obviously, Geometric Range Assignment instances on the real line $\mathbb{R}$ are linear. Although allowing quite "strange" distance functions (we will see an example later on), this notion makes sense insofar as it essentially covers what makes the dynamic program of [KKKP00] for Geometric Strong Connectivity work. To put it clearly: Linear Range Assignment Problems allow a polynomial time exact algorithm.

Now to the construction of our linear example family whose members we call $L_{j,\varepsilon}$. They are somewhat similar to instances $I_{k,\varepsilon}$. We define

$$\begin{aligned} L_{k,\varepsilon} \;=\; & \{0, \varepsilon; \\ & 1-\varepsilon, 1, 1+\varepsilon; \\ & 2-\varepsilon, 2, 2+\varepsilon; \\ & \ldots; \\ & (k-1)-\varepsilon, k-1, (k-1)+\varepsilon; \\ & k-\varepsilon, k\} \end{aligned}$$

Neighboring points in fact have their normal Euclidean distance. The more curious distances are as follows:

$$\begin{aligned} d(i-1, i-\varepsilon) &= 2 \qquad \forall\ 1 \leq i \leq k \\ d(i+\varepsilon, i+1) &= 2 \qquad \forall\ 0 \leq i \leq k-1 \end{aligned}$$

All other distances are the sum of edge weigths of a path *with least number of edges* on the above defined edges. By construction, the distance function is linear, with the natural order.

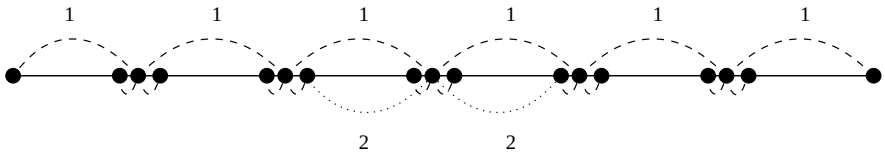

Figure 4.6: With some adjustment of the Euclidean distances, factor 2 is tight on the line.

Maybe the reader her-/himself would like to try out the Prim or Kruskal greedy algorithm on this kind of instance. You will discover that both algorithms will actually come up with an MST, so that ranges on points $i \pm \varepsilon$ are large

$(1 - 2\varepsilon)$. In an optimal solution, only integer points i have radius 1, only half as many. This is because the "good" integer stations i are hidden from left and right by stations $i \pm \varepsilon$, which produce much worse range assignment although they lie so close to good station due to the "curious"—but still monotone—distances.

Observation 4.8. *Approximation factor 2 for greedy heuristics* Kruskal *and* Prim *is tight on linear instances.*

□

A geometric example for Kruskal

It is unclear how to obtain a geometric tight example in $\mathbb{R}$, so the next thing to try is $\mathbb{R}^2$, where obviously much more complex instances are possible than in $\mathbb{R}$ (in particular, NP-hard ones). However, our examples are still inspired by the linear one above, at least in principle.

The first example we present shows tightness only for Kruskal, but not the Prim heuristic. Our second example will work for both algorithms. The reason why we include this example as well is that (a) it is somewhat simpler and (b) highlights differences between Kruskal and Prim.

This time, we will not give full details of all stations, but rather a picture which should convince the reader as well. One reason is that we make use of Lemma 2.25 again, so we have to add a huge number of extra points, depending on the power-distance gradient α, to approach factor 2.

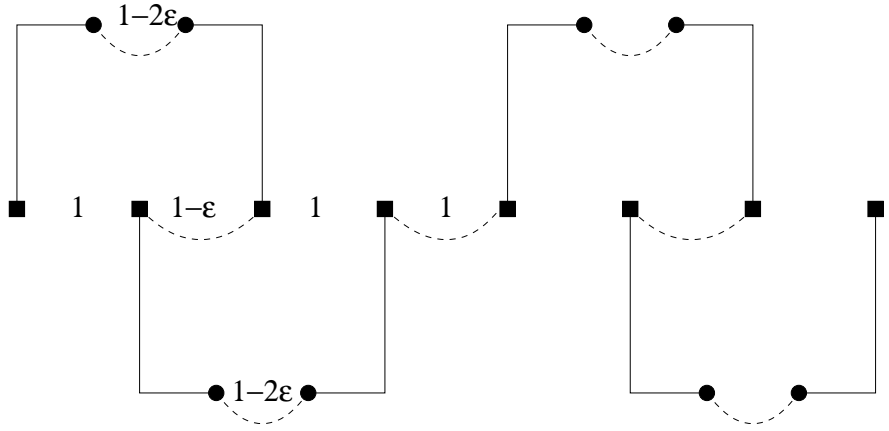

Figure 4.7: A building block of our tightness example for Kruskal.

Figure 4.7 shows one building block for our first example. Larger instances are constructed by placing the left square of a new building block at distance one to the right of the rightmost square of an old instance. Circles and squares indicate points of interest. On the lines, points are placed at some (small) distance δ,

such that the power used for communication along them can be neglected. At this point, we need that distance-power gradient $\alpha > 1$. By adding more blocks, we come closer to factor 2; also by decreasing parameters ε and δ.

Now consider algorithm KRUSKAL on such an instance. At first, every point is assigned a range δ, connecting the points on the lines. Then, all $(1-2\varepsilon)$ edges are added, thus all circled points will have range about 1. After that, the $(1-\varepsilon)$ edges are included, and finally some of the length 1 edges edges, which will cause all squared points to have range about 1 (except for the leftmost and rightmost point). In an optimal range assignment, we only need all squared points to have range 1. There are as many circled points as squared points, so we have about twice as many range 1 points in a KRUSKAL solution (which is, by the way, identical to the MST solution) as in an optimal solution. The total power cost of range δ nodes can be made arbitrarily small due to Lemma 2.25. We conclude:

Observation 4.9. *Approximation factor 2 of algorithm* KRUSKAL *is tight for* GEOMETRIC (STRONG) CONNECTIVITY *in* $\mathbb{R}^2$ *and* $\alpha > 1$.

□

Algorithm PRIM is not fooled by this example. Consider the instance in Figure 4.7, and start PRIM on the left square node. It moves along the incident line, adds the upper $(1-2\varepsilon)$-edge, and the $(1-\varepsilon)$-edge (and incident δ-lines). The next choice is between the lower $(1-2\varepsilon)$-edge and a length 1 edge. This time, PRIM will select the length 1 edge as one node already has radius $(1-\varepsilon)$, making it cheaper to add than the $(1-2\varepsilon)$-edge. The algorithm will continue giving the squared stations range 1, which is the optimal choice.

A geometric example for PRIM

Again, we explain our example by a picture of one building block which are connected horizontally to give instances of any desired size.

This time, the left square of the next block is placed directly on top of the rightmost square of a smaller example. Two neighboring circled points have distance $1-\varepsilon$, two neighboring squared points have distance 1. Apart from the δ-lines, an MST uses all $(1-\varepsilon)$-edges, while an optimal range assignment uses the length 1 edges between the squared points. When we use more and more building block and decrease ε and δ, the MST range assignment comes arbitraliry close to twice the cost of an optimal range assignment. The reader may want to check that both the KRUSKAL and PRIM greedy heuristic will compute an MST on this kind of instance, showing

Observation 4.10. *Approximation factor 2 of algorithm* PRIM *is tight for* GEOMETRIC (STRONG) CONNECTIVITY *in* $\mathbb{R}^2$ *and* $\alpha > 1$.

□

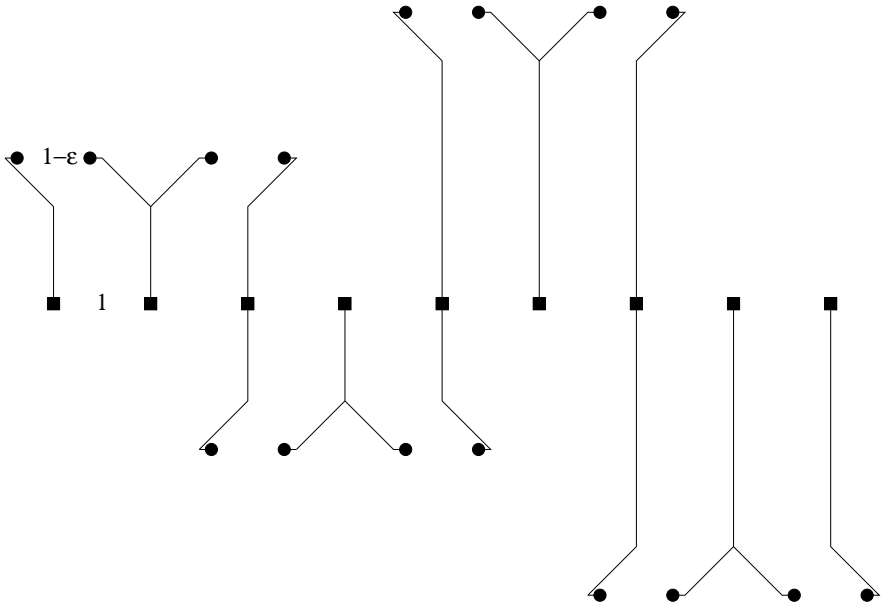

Figure 4.8: One building block for the tightness example in $\mathbb{R}^2$.

Metric examples

The above geometric tight instances will not work any longer when $\alpha = 1$, as we cannot neglect the overhead costs on the lines any longer. While we doubt that there will be tight examples in $\mathbb{R}$ or $\mathbb{R}^2$ with Euclidean distances, we will see now that for general metric instances, factor 2 is also tight already.

We will first show an example fooling Prim. It needs some further technical modifications to work for Kruskal, too. The instance basically consists of a star with k leaves at distance $1 + \varepsilon$ to center node c. Additionally, a length 1 path of "very many very close" stations leads to c. The other end of this path, t, has a length 2 edge connected to a vertex s, on which the Prim algorithm will be started. We add a tour through the star leaves and t, where all edges have weight 1, see Figure 4.9.

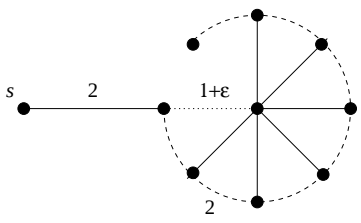

Figure 4.9: A tight instance for Prim...

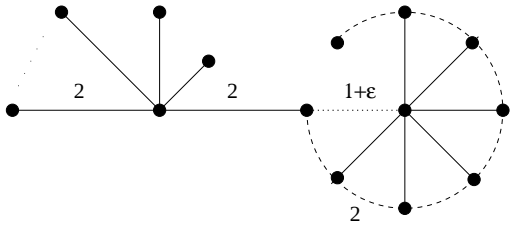

Figure 4.10: ... and an adaption for Kruskal.

As usual, the distance between two points is the length of a shortest path in

the above graph. Now consider the PRIM algorithm started on s. It will first insert $\{s,t\}$, assigning radius 2 to node t. After the small edges on the length 1 path to c are inserted, the important thing is that the first star leaf is not attached to the star center c via the length $(1+\varepsilon)$ edge at cost $2+2\varepsilon$ but to t via the length 2 edge at cost 2. The same holds for the next leaf on the tour, and so on. Thus we end up with all k leaves at radius 2 and the central node a negligible radius, while in an optimal range assignment (which is in fact the MST range assingment), all leaves and the center have radius 1. With $k \to \infty$, the ratio between the two solutions comes arbitrarily close to 2.

The example does not work as is for KRUSKAL, as the leaves would be connected to c before $\{s,t\}$ is added finally, resulting in an optimal solution. We have to get the range of t close to 2 in another way. Instead of $\{s,t\}$, we add edges $\{s_i,t\}$ with $w(s_i,t) = 2-(\frac{1}{2})^i$ for $0 \le i \le \log_2(1/\varepsilon)+1$, cf. Fig. 4.10. In this way, the range t comes close enough to 2, and KRUSKAL will also eventually follow the tour through the star leaves. Notice however that we will need far more star leaves in this reduction in order to compensate the additional initial star around t.

These two examples show

Theorem 4.11. *Approximation factor 2 of algorithms* KRUSKAL *and* PRIM *is tight for metric* (STRONG) CONNECTIVITY.

Notice however that we cannot embed these examples into Euclidean spaces, as the more star leaves we try to embed, the closer they will be to each other. We thus conjecture that the greedy algorithms have an approximation ratio which is strictly lower than 2 in Euclidean R^d, and dependent on d.

4.2.4 Greedy algorithms and BROADCAST

In this section, we just state some thoughts and ideas about a greedy algorithm for GEOMETRIC BROADCAST. In the BROADCAST context, the strategy in Figure 4.11 is very intuitive:

Notice a subtle difference to algorithm PRIM for (STRONG) CONNECTIVITY: As we add arcs (v,w) instead of edges, these only add to the range of their outgoing station v. Thus, *cost* differs in PRIM' from the *cost* in PRIM, which we tried to indicate by writing Δ' instead of Δ. This way, nodes with already larger range are preferred even more than in the (STRONG) CONNECTIVITY context, which seems quite plausible for BROADCAST.

Observation 4.12. *If ties are broken in favor of the node with largest radius, algorithm* PRIM' *works optimally on* GEOMETRIC BROADCAST *with* $\alpha \le 1$.

Proof. Due to the $\triangle$-inequality and tie breaking rule, the PRIM' solution will always be a star around s, which is optimal for metric BROADCAST. □

Algorithm PRIM' **for** BROADCAST

Instance: A weighted graph $G = (V, E, d)$ with distances $d : E \to \mathbb{R}_+$, and a source station $s \in V$.

Output: An arborescence P rooted at s spanning G.

Algorithm:

- **Let** $i := 0$, $V_0 := \{s\}$, $P_i := \varnothing$.
- **Repeat**
  - **Let** $i := i + 1$
  - **Choose** $p_i = (v, w) \in E$ s.t. $v \in V_{i-1}$, $w \notin V_{i-1}$, and p_i minimizes
$$\Delta'(p) := cost(P_{i-1} \cup \{p\}) - cost(P_{i-1}).$$
  - **Let** $P_i := P_{i-1} \cup \{p_i\}$, $V_i := V_{i-1} \cup \{w\}$.
- **Until** $i = n - 1$.
- **Output** $P := P_{n-1}$.

Figure 4.11: Algorithm PRIM' FOR BROADCAST.

Although we do not need good algorithms for trivial cases, the above observation does not hold for the MST-heuristic, the best known algorithm for GEOMETRIC BROADCAST. So what happens when α becomes greater than one, but only slightly? We do not know, but chances are that PRIM' does not already become completely hopeless immediately. So it is a candidate for filling the gap of $1 < \alpha < 2$, for which no constant factor approximation for BROADCAST in $\mathbb{R}^2$ is known yet. It shows a nice behavior on the grid $\mathbb{Z}^2$, where MST fails for $\alpha < 2$: Its solution structure changes exactly at $\alpha = 2$, where it computes an MST for $\alpha > 2$ and a star for $\alpha \leq 2$, a behavior suggested by the calculations for the MST on the grid. (We will elaborate on this in a few paragraphs.) Moreover, it performs optimally on the worst-cast example for MST when $\alpha \geq 2$, so it may be superior to MST also in this case. There is a lower bound of $13/3 \approx 4.33$ on its performance for $\alpha = 2$.[WaLF02] What we actually know is that it is at least as good as MST:

Observation 4.13. *For $\alpha \geq 2$,* PRIM' *is at least a 6-approximation for* BROADCAST *in $\mathbb{R}^2$.*

Proof. We can assume that the cost of an MST is about the same as $|MST|$: By adding ε-points around each point like in Figure 4.3, every arc longer than ε

dominates a vertex. On the other hand, the same proof as for Theorem 4.6 shows that a PRIM' solution costs at most $|MST|$, so the approximation ratio of PRIM' is no worse than that of MST. □

So we know that PRIM' works optimally for $\alpha \leq 1$ and is a constant factor approximation for $\alpha \geq 2$. But even if PRIM' would perform well for some small $\alpha > 1$, it is no universal algorithm for all $\alpha > 0$. We can show that the gap of $1 < \alpha < 2$ cannot be closed completely.

Theorem 4.14. *Algorithm* PRIM' *does not yield a constant factor approximation for* GEOMETRIC BROADCAST *for any* $\beta < \alpha < 2$*, where* $\beta = 2\log_3 2 \approx 1.26$.

Proof. As mentioned above, PRIM' switches at the right time from an MST to a star solution, namely at $\alpha = 2$. This is different in another kind of grid, which is more dense. This grid consists of equilateral triangles instead of squares, and reminds of a honeycomb, which is how we will call our instances.

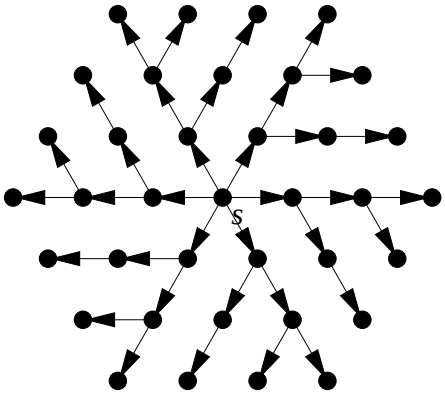

Figure 4.12: The 3-comb, with an MST range assignment.

Definition 4.15. For an integer $k \geq 1$, we call the vertices of a triangulation of a hexagon with sidelength k with equilateral triangles of sidelength 1 a *k-comb*. The central node is source node s.

See Figure 4.12 for a 3-comb. When s transmits directly to all stations, this costs k^α. An MST may assign all nodes but the outer ones radius 1, see Figure 4.12. There are $3k^2 + 3k + 1$ nodes in a k-comb, so this would give a range assignment of cost $3k^2 - 3k + 1 = \Theta(k^2)$. Thus, an MST is no constant factor approximation for $\alpha < 2$. For which values of α will PRIM' construct a range assignment like the one in Figure 4.12? When we start PRIM' at s, the six incident length 1 arcs are added. The next arc outgoing from s has length $\sqrt{3}$, adding which would cost $\sqrt{3}^\alpha - 1$. The same node could be reached from two of the other nodes around s at cost 1, see Figure 4.13.

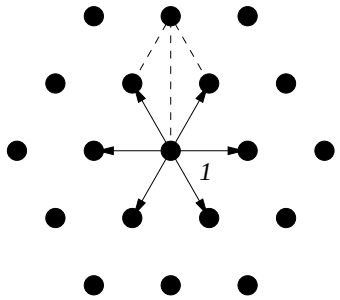

Figure 4.13: To connect the northern point, PRIM' has to decide between two length 1 arcs and a $\sqrt{3}$ arc.

So for

$$\sqrt{3}^{\alpha} - 1 > 1 \iff 3^{\frac{\alpha}{2}} > 2 \iff \alpha > 2\log_3 2$$

PRIM' produces the MST-solution on a k-comb, which is no constant factor approximation for $\alpha < 2$. □

It is quite remarkable that PRIM' has this kind of gap for α regarding constant factor approximation. We conjecture that PRIM' approximates BROADCAST in $\mathbb{R}^2$ strictly better than 6 for $\alpha = 2$, and is a constant factor approximimation for $1 < \alpha \leq \gamma$ for some $1 < \gamma < \beta$.

4.3 Limitations of the MST lower bound

How can we get beyond approximation ratio 2 for (STRONG) CONNECTIVITY, maybe only for certain types of instances, e.g. metric ones? In the analysis of the greedy algorithms KRUSKAL and PRIM we conjectured that these algorithms perform better in a Euclidean setting. One suggestive approach for such a result for, say, algorithm PRIM, would be as follows. Let P be the solution of algorithm PRIM, and MST an MST. It would suffice to show that $cost(P) < \beta \cdot |MST|$ for some $\beta < 2$ for all Euclidean instances. This is what we did for all factor 2 algorithms. However, we can show that this kind of approach will not work. We construct a family of Euclidean instances already on the real line $\mathbb{R}$ where an optimal range assignment comes arbitrarily close to $2|MST|$. Thus, lower bound $|MST|$ is already fully exploited for approximating CONNECTIVITY, and we need a better lower bound for improved results.

Theorem 4.16. *For every $\varepsilon > 0$, there is an instance on the real line $\mathbb{R}$ with Euclidean distances with*

$$cost(OPT) > (2 - \varepsilon)(|MST| + \max_{e \in MST} |e|),$$

where OPT is an optimal CONNECTIVITY *range assignment.*

(Without the additional term $\max_{e \in MST} |e|$, a single edge would trivially show the above statement.)

Proof. Again, our instances are made up of building blocks which we describe by a picture. Only in this case, the building blocks are not arranged next to each other, but have to be inserted into one another in a recursive manner.

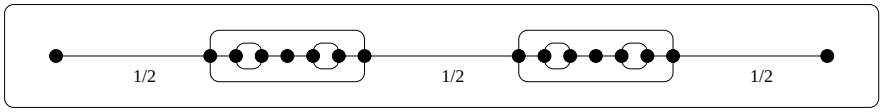

Figure 4.14: The recursion used to build up our instances.

The first level of our recursion just consists of some (large) number of points at distance 1/2. Recursively, insert such an instance (appropriately scaled, of course) into *every other* free space between two points. In every such inserted instance of equally spaced points, again insert a newly scaled instance into every other free space, and so on.

On the first level, we need one point with range one per building block to connect all the building blocks. This means we already need a range assignment of cost the size of the interval, say k, our example lives in, just to connect the k building blocks. In all subsequent levels, the same argument holds with respect to scaling. On the second level, the intervals occupied by building blocks add up to $k/2$. On the third level, it is $k/4$, etc. In total, when we apply this recursion more and more times, the optimal range assignment for the constructed instance approaches $k(1 + 1/2 + 1/4 + \ldots) = 2k$, while a minimum spanning tree for this instance has size k. So for every $\varepsilon > 0$, an appropriate choice of k and a finite number of recursions of the above construction produces an instance where $cost(OPT) > (2 - \varepsilon)(|MST| + \max_{e \in MST} |e|)$, proving our claim. □

4.4 Metric Strong Connectivity

4.4.1 Using Connectivity algorithms for Strong Connectivity

We make the following observation about general approximation of Strong Connectivity by Connectivity algorithms.

Theorem 4.17. *Let OPT_C be an optimal* Connectivity *solution, and OPT_{SC} an optimal* Strong Connectivity *solution on some fixed range assignment instance. For every $\varepsilon > 0$, there is a range assignment instance in $\mathbb{R}^2$ with Euclidean distances for which*

$$cost(OPT_C) > (2 - \varepsilon) cost(OPT_{SC})$$

holds.

Proof. The proof is based on Theorem 4.16, which is not too surprising, as the two values are nested in the following way.

$$|MST| \leq cost(OPT_{SC}) \leq cost(OPT_C) \leq 2 \cdot |MST|$$

So we need instances where $cost(OPT_C) \approx 2|MST|$, and $cost(OPT_{SC}) \approx |MST|$. We have constructed instances with the first property in Theorem 4.16. Instances with the latter property are, e.g., stars with many leaves, and in $\mathbb{R}^2$ with Euclidean distances, instances $\mathbb{Z}_k^2$ for large k.

We thus construct 'grids' of instances as in Theorem 4.16 as follows. We scale such an instance so it has total length 1, and put one such unit instance between every pair of horizontal or vertical neighbors, see Figure 4.15.

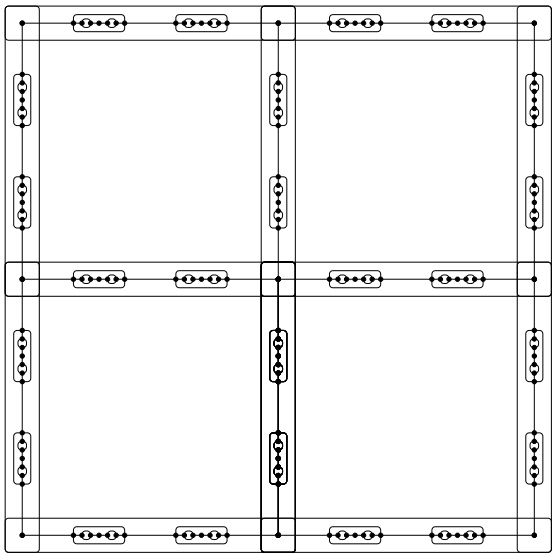

Figure 4.15: Instances as in Theorem 4.16, arranged on grid $\mathbb{Z}_1^2$.

An optimal Connectivity range assignment will be identical to the optimum on all individual instances, with cost arbitrarily close to $2|MST|$. An optimal Strong Connectivity range assignment will be an MST directed towards the star center, and a radius of $\sqrt{2}k$ on the star center (in fact, this is the solution of the 'all-to-one, one-to-all' heuristic). We have that $|MST| = 4k^2$, and thus for $cost(OPT_{SC})$:

$$cost(OPT_{SC}) = |MST| + \sqrt{2}k = |MST|(1 + \frac{\sqrt{2}k}{4k^2}) = |MST|(1 + \frac{1}{k})$$

So with increasing k, we get the ratio $cost(OPT_C)/cost(OPT_{SC})$ arbitrarily close to 2. □

Being interesting by itself, as it shows how much the two connectivity notions can differ, this Theorem immediately yields the following Corollary regarding approximations for the two problems.

Corollary 4.18. *An algorithm for* CONNECTIVITY *cannot be better than a 2-approximation for* STRONG CONNECTIVITY*, already on instances in* $\mathbb{R}^2$ *with Euclidean distances.*

Regarding this fact, the MST heuristic and the greedy heuristics are 'optimal' in some sense, as even taking an optimal CONNECTIVITY range assignment would give no better approximation of STRONG CONNECTIVITY.

4.4.2 Purpose built STRONG CONNECTIVITY algorithms

Let us come back to the example in Theorem 4.16 showing that an optimal CONNECTIVITY range assignment can cost as much as $2 \cdot |MST|$, showing that no better approximation ratio than 2 can be shown by this lower bound for metric cases. This example does not work in the STRONG CONNECTIVITY case. In fact, the recursive construction is unnecessary; the same effect is achieved just by taking the usual "lots of close stations" on every other gap between two stations. An optimal strongly connected range assignment has one 'large' radius of 1.5 in the middle of each cluster of 'close' stations, and cost 1 on all other stations in this cluster, sending to the middle station. (See Figure 4.16 in the next Theorem for an illustration of an optimal range assignment.) These total costs of 2.5 per cluster/gap combination stand against MST length of 2, showing a lower bound of 5/4 altogether for the ratio $cost(OPT)/|MST|$ for STRONG CONNECTIVITY on the Euclidean line. We get an improved lower bound in this kind of gap/cluster setting for a different choice of parameters.

Theorem 4.19. *For every* $\varepsilon > 0$*, there is a range assignment instance on the real line* $\mathbb{R}$ *with Euclidean distances with*

$$cost(OPT) > \left(\frac{4}{3} - \varepsilon\right)(|MST| + \max_{e \in MST} |e|),$$

where OPT is an optimal STRONG CONNECTIVITY *range assignment.*

Proof. Figure 4.16 shows our instances, together with an optimal range assignment. Our instances consist of clusters of length 2 with lots of close (distance δ) points, at distance 1 to the next clusters left and right. An optimal range assignment has radius 2 on the middle point in each cluster, and δ on all other points, resulting in cost 4 per cluster. Each cluster/gap combination adds length 3 to an MST, showing lower bound 4/3, as desired. □

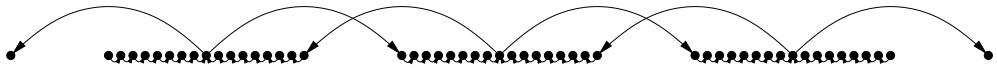

Figure 4.16: An optimal Strong Connectivity range assignment with cost $4/3 \cdot |MST|$.

Note that on this example, the MST heuristic also has cost 4 per cluster (cost 1 on either end, and total cost 2 in between), which is the reason for this specific example giving the best lower bound among the gap/cluster kind of examples.

On the other hand, the above Theorem gives a lower bound quite far away from the upper bound 2 we know, meaning there is hope to prove something better than 2 for metric Strong Connectivity with the $|MST|$ lower bound. However, we have not yet ruled out that this might be possible even for non-metric instances, i.e. Geometric Strong Connectivity with $\alpha > 1$. But we cannot hope for this, as the same example as above shows in a geometric setting.

Corollary 4.20. *For every $\varepsilon > 0$, there is a* Geometric Strong Connectivity *instance in $\mathbb{R}$ with $\alpha > 1$ with*

$$cost(OPT) > (2 - \varepsilon)(|MST| + \max_{e \in MST} |e|),$$

where OPT is an optimal Strong Connectivity *range assignment.*

Proof. Consider the instance in 4.16 above again, this time with $\alpha > 1$. The difference is that we can once more make use of Lemma 2.25 and may neglect the power consumed by the small radius stations on the lines. The MST-heuristic is optimal, having cost $1^\alpha = 1$ at the leftmost and rightmost station of each cluster, and negligible power cost inside the cluster, adding up to cost of about 2 per cluster. An MST has a length 1 edge on each gap, and negligible edges inside the cluster, so in total an MST has only cost of about 1 per cluster/gap combination, showing that the ratio $cost(OPT)/|MST|$ can come arbitrarily close to 2 on these instances. □

Back to the metric case, where we know of no better lower bound for the ratio $cost(OPT)/|MST|$ than 4/3. Remember the 'one-to-all, all-to-one' algorithm from Chapter 2 which proved useful (i.e., provided PTASs) in some restricted settings of Strong Connectivity? It is also an improved approximation algorithm for metric Strong Connectivity, which has already been observed by Ambühl *et al.* in [ACP+04].

Theorem 4.21 ([ACP+04]). *Algorithm 'All-to-one, one-to-all' is a 3/2-approximation algorithm for metric* Strong Connectivity.

Proof. Algorithm 'all-to-one, one-to-all' directs an MST towards a hub node s, which is chosen as to minimize

$$R = \min_{v \in S} \max_{w \in S} d(v, w),$$

i.e., s is the node with the least maximal distance R to any other node. In some sense, s is the 'center' of G, and R is its 'radius'.

So the algorithm's solution T will have cost

$$cost(T) = |MST| + R.$$

The following lemma is central to the proof of this Theorem.

Lemma 4.22. *For a graph G with metric distances d and minimum spanning tree MST, we have*

$$R \leq \frac{|MST| + w}{2}$$

where $w = \max_{e \in MST} |e|$.

Proof. We give a constructive proof for this Lemma: we identify a point m which has maximal distance to any other point at most as claimed above.

Let P be a longest path in MST, with endpoints a and b. Imagine this path put on the real line $\mathbb{R}$, with point a at 0 and b at $|P|$. The exact middle point of this path lies at $|P|/2$; pick the vertex closest to this arithmetic mean and call it m. Cf. Fig. 4.17.

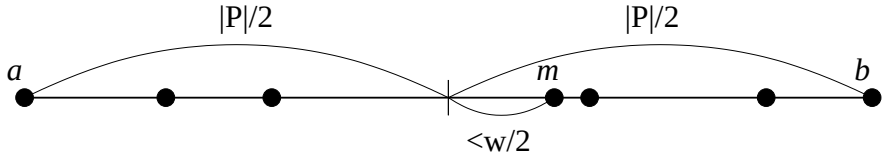

Figure 4.17: Path P, with arithmetic mean and closest station m.

Now m lies at most $w/2$ from the arithmetic mean, and thus the $\triangle$-inequality assures it has total distance at most $|P|/2+w/2$ to the further of the two endpoints of P, and thus in fact to *any* point in G (else P would not have been a longest path). Of course, $|P| \leq |MST|$, which closes the proof of this Lemma. □

We now calculate

$$\begin{aligned} \frac{cost(T)}{cost(OPT)} &\leq \frac{|MST| + R}{|MST| + w} \leq \frac{\frac{3}{2}|MST| + \frac{1}{2}w}{|MST| + w} \\ &= \frac{3}{2} - \frac{w}{|MST| + w} \leq \frac{3}{2} - \frac{1}{n} \end{aligned}$$

showing the Theorem. (Actually, we have a $(3/2 - 1/n)$-approximation.) □

This analysis is tight, as instances $\mathbb{Z}_k := \{z \in \mathbb{Z} \mid |z| \leq k\}$ show: Additionally to having radius 1 on each point, radius $r(0) = k$ is unnecessarily high. We have $cost(OPT) = 2k + 1$ and $cost(T) = 3k$, showing tightness asymptotically.

The same sets of points show that 'all-to-one, one-to-all' is no constant factor approximation in the non-metric case. Already for Geometric Strong Connectivity in $\mathbb{R}$ for any $\alpha > 1$, on instances $\mathbb{Z}_k$ we observe the following behavior. Optimal costs $cost(OPT) = 2k + 1$ remain the same, but the algorithm's costs $cost(T) = 2k + k^\alpha$ increases, and the ratio between the two expressions cannot be bounded by a constant anymore.

4.4.3 Well-spread instances

Although we have just seen that 'all-to-one, one-to-all' is no constant factor approximation already in a very simple geometric setting, it deserves a closer examination on well-spread instances. Having said that, instances $\mathbb{Z}_k$ form *the* prototype of a well-spread family in $\mathbb{R}$, where the algorithm fails for any $\alpha > 1$. But it has also been observed in [ACP+04] that it does yield a constant factor approximation for well-spread instances in $\mathbb{R}^2$ for $\alpha = 2$, albeit dependent on the (constant) parameter c in the definition of well-spreadness, cf. Def. 3.2.

This dependence makes 'all-to-one, one-to-all' in the above $\mathbb{R}^2$ case obsolete, as the MST-heuristic has performance guarantee 2 on whatever instance. However, a more detailed analysis reveals that it can be of more use in other cases.

Theorem 4.23. *On well-spread instances with parameter c of* Geometric Strong Connectivity *in $\mathbb{R}^d$ with distance-power gradient α, algorithm 'all-to-one, one-to-all' has the following performances, in dependence of d and α:*

1. *For $d < \alpha$, it is no constant factor approximation.*
2. *For $d = \alpha$, it is a constant factor approximation, where the constant factor depends on c.*
3. *For $d > \alpha$, it gives rise to a PTAS.*

Proof. The algorithm's cost amounts to $cost(T) = |MST| + R$, as above. The instance S is assumed to be well-spread, with maximal (Euclidean!) distance $\Delta = \Delta(S)$ and minimal distance $\delta = \delta(S)$. This property yields $R \leq \Delta^\alpha$ and $\delta \geq c\Delta/\sqrt[d]{n}$. Together with $|MST| \geq (n-1)\delta^\alpha$, we have the following line of inequalities.

$$\begin{aligned}
\frac{cost(T)}{cost(OPT)} &\leq \frac{|MST| + R}{|MST| + w} \leq 1 + \frac{R}{|MST| + w} \\
&= 1 + \frac{1}{n} \cdot \left(\frac{\Delta}{\delta}\right)^\alpha \leq 1 + \frac{1}{n} \cdot \left(\frac{\sqrt[d]{n}}{c}\right)^\alpha \\
&= 1 + \frac{1}{c^\alpha} \cdot n^{\frac{\alpha}{d} - 1}
\end{aligned}$$

For $\alpha > d$, the last term is unbounded for increasing n. (Being an upper bound only, this does not yet prove statement 1.; we will come back to this later.)

For $\alpha = d$, we have

$$\frac{cost(T)}{cost(OPT)} \leq 1 + \frac{1}{c^\alpha} = O(1),$$

i.e., the algorithm is a constant factor approximation in this case, showing statement 2.

For $\alpha < d$, define $\beta = 1 - \frac{\alpha}{d} > 0$. We have

$$\frac{cost(T)}{cost(OPT)} \leq 1 + \frac{1}{c^\alpha} \cdot \frac{1}{n^\beta}.$$

We want to obtain a $(1+\varepsilon)$-approximation. For $\frac{1}{c^\alpha} \cdot \frac{1}{n^\beta} \leq \varepsilon$, we are done already. So we assume this is not the case, implying

$$1 + \frac{1}{c^\alpha} \cdot \frac{1}{n^\beta} > \varepsilon \iff n < \sqrt[\beta]{\frac{1}{c^\alpha \varepsilon}} = O(1).$$

Thus there is a constant upper bound on n up to which we can try out all possible range assignments in time $O(1)$; for bigger instances, the algorithm 'all-to-one, one-to-all' will be a $(1+\varepsilon)$-approximation, showing statement 3.

We come back to the case $\alpha > d$. Consider instances

$$\mathbb{Z}_k^d = \{z \in \mathbb{Z}^d \mid |z_i| \leq k\},$$

d-dimensional grids lying inside a hypercube with side-length $2k$ around the origin. $(\mathbb{Z}_k^d)_k$ is well-spread in $\mathbb{R}^d$. A minimal configuration, where every node has range 1, is feasible and thus optimal, at cost

$$cost(OPT) = (2k+1)^d = \Theta(k^d).$$

The algorithm additionally increases the origin's radius to $(\sqrt{d} \cdot k)^\alpha$, so we have

$$cost(T) = (2k+1)^d + (\sqrt{d} \cdot k)^\alpha = \Theta(k^\alpha),$$

implying

$$\frac{cost(T)}{cost(OPT)} = \Theta(k^{\alpha-d}),$$

which not bounded by a constant, implying statement 1. □

So there exists a PTAS for well-spread instances, e.g. in the Euclidean plane, or in fact for any $\alpha < 2$. For dimension $d = 3$ and higher, we now have nearly the full picture of the PTAS approximability of well-spread STRONG CONNECTIVITY: For $\alpha < d$ we have a PTAS, and for $\alpha > d$ there cannot be one unless P = NP.

4.4.4 A new algorithm for metric Strong Connectivity

In this section, we present a new approximation algorithm for Strong Connectivity. While we cannot prove a better approximation ratio for metric cases than for 'all-to-one, one-to-all', it has some interesting features we will discuss afterwards. We present our algorithm on the real line first, and generalize it to arbitrary instances. We call our algorithm *'Combing back and forth'*, or just *'Combing'*.

Combing the real line

Let $S = \{s_1, \ldots, s_n\}$ a Strong Connectivity instance on the Euclidean real line, with $s_1 < s_2 < \ldots < s_n$. In the first step, we 'comb' the line from right to left. I.e., we set each radius $r(s_i) = d(s_{i-1}, s_i)$, for all $1 < i \leq n$. This is equivalent to inserting the arcs (s_i, s_{i_1}), i.e., an MST with all arcs pointing left.

In the second (and final) step, we comb back, from left to right. By this we mean the following. The range assignment after the first step naturally ensures a directed path for each node to every node on its left, while it already contains some arcs to the right as well. We insert missing arcs resp. increase arc lengths from left to right to ensure paths to each station on the right as well, implying overall strong connectivity.

The algorithm is formulated in Figure 4.18. We can view this algorithm as a directed counterpart to the preceding greedy algorithms as it iteratively adds arcs to a range assignment in a greedy fashion until it is strongly connected. The following proof of the approximation factor is, in our opinion, quite nice and simple.

Theorem 4.24. *Algorithm 'Combing' is a 2 approximation algorithm for* Strong Connectivity *on the line (i.e., general linear instances). On metric distances on the line, it achieves an approximation ratio of* 1.5*.*

Proof. Both proofs again rely on the MST lower bound in Lemma 1.11. After the first combing step, we obviously have $cost(C_1) = |MST|$. In the second step, when arc (s_k, s_j) is added, we have considered adding arc (s_{j-1}, s_j). This means that adding all MST edges as arcs pointing right is an upper bound for the second step, showing a total upper bound of $cost(C) \leq 2 \cdot |MST|$ for our algorithm, showing the first claim.

Now assume our distances are metric. What we do in step 2 of our algorithm is to fill the gap between s_{j-1} and s_j, be it with arc (s_{j-1}, s_j) or by increasing some other radius by this length. By the $\triangle$-inequality, this costs at most $d(s_{j-1}, s_j)$. Now observe that this means that station s_j is already assigned range $d(s_{j-1}, s_j)$ from the first step! This means for connecting the next point $s_{j'} > s_j$, we have to pay at most an additional $d(s_j, s_{j'}) - d(s_{j-1}, s_j)$, and so forth. To put it plainly, in the second step, we are in a constant 'buy one, get one free' situation: For

Algorithm 'Combing' for STRONG CONNECTIVITY **in** $\mathbb{R}$

Instance: Stations $S = \{s_1, \ldots, s_n\}$ with $s_1 < s_2 < \ldots < s_n$.

Output: A strongly connected network C.

Algorithm:

- **Let** $i := 1$, $C_1 := \{(s_j, s_{j+1}) \mid 1 \leq j < n\}$.
- **Repeat**
  - **Let** s_j be the station to which there is no directed path from s_i, with j minimal.
  - **Choose** station s_k with $k < j$ minimizing

$$\Delta(k) := cost(C_i \cup (s_k, s_j)) - cost(C_{i-1})$$

  - **Let** $C_j := C_i \cup (s_k, s_j)$, $i \leftarrow j$.
- **Until** C_i is strongly connected.
- **Output** $C := C_i$.

Figure 4.18: Algorithm Combing in $\mathbb{R}$.

every new arc unit that we buy here, we know that we actually can transmit twice that distance to the right. This is always true but for the last arc that is added: The 'travel bonus' may now lie beyond our instance and be without use for us. However, this is covered by the additive term $\max_{e \in MST} |e|$ in our lower bound from Lemma 1.11.

Apart from this last edge, we only add at most half of $|MST|$ in the second step. So our total range assignment has cost at most

$$cost(C) \leq \frac{3}{2}|MST| + \max_{e \in MST} |e|,$$

proving our second claim. □

To see that factor 2 is tight for general linear instances, we can again use instances $L_{j,\varepsilon}$, as in Figure 4.6. It is also not difficult to see that the above bounds are tight in terms of the MST lower bound. Indeed, this is clear a priori for geometric instances due to Corollary 4.20. For the metric case on the line, consider instances consisting of stations at unit distance, and lots of stations in every other gap. Here, the range assignment from the first sweep only grants us no more than $|MST|/2$ worth of 'free rides' to the right, and the 'Combing' range assignment will indeed cost up to $\frac{3}{2}|MST| + \max_{e \in MST} |e|$.

However, this does not mean that the approximation ratio of this algorithm is tight, neither for geometric nor metric instances. In the above example, an optimal range assignment looks as follows. The middle station in each 'crowded' gap has radius 1.5, while the other stations in this gap send to this hub station at cost 1. This means total costs of 2.5 on each crowded gap, compared to costs of 3 in the Combing solution. This means the above instance merely yields a lower bound of 1.2 on the approximation ratio for 'Combing' on the metric real line. We know of no better lower bound.

For geometric cases on the line, the same instances as above give a slightly better lower bound of 4/3, but compared to our upper bound of 2, this still leaves us with a huge gap.

Let us briefly summarize these results.

Lemma 4.25. *The approximation ratio of the algorithm 'Combing' the real line cannot be better than*

- 2 *for general linear instances,*
- 4/3 *for geometric instances, and*
- 6/5 *for metric instances.*

□

So there is still hope that our new Combing algorithm may actually outperform 'all-to-one, one-to-all' on the metric line. Another thing that makes it superior is that Combing does not immediately become completely hopeless on geometric instances for any $\alpha > 1$. 'All-to-one, one-to-all' does, even on the arguably most simple instances. Here, Combing computes an optimal solution, and it does provide us with at least a 2-approximation for general geometric instances, maybe even better.

A further remark is a practical one. A typical solution computed by 'all-to-one, one-to-all' heavily relies on the central hub node. Its energy consumption will probably be much higher than that of any other node, and most communications will use this node. This may well cause the hub node to become a bottleneck for the overall network activity, slowing communication down unnecessarily. Furthermore, this even makes the network more vulnerable, as battery supplies in the hub node may run out quickly, and a breakdown of the hub node will cause a mass breakdown of nearly all communication. The solutions of Combing, on the other hand, will have a more local structure, and avoid the above disadvantages of a single hub node. But as our work is of more theoretical nature, we merely leave this as a side note. Realistic networks will probably use an $\alpha \geq 2$, which already disqualifies the 'all-to-one, one-to-all' heuristic as a reasonable approximation algorithm.

Of course, we would like an algorithm that does not only work on instances in $\mathbb{R}$. In the next section, we generalize Combing to general metric instances.

Combing general metric instances

The idea of the Combing algorithm does make essential use of the path structure in $\mathbb{R}$. What we do to transfer this idea to general metric instances is to first compute an MST in such an instance, and basically apply Combing to a *longest path* in the MST.

Let $s_1, s_2, \ldots, s_l$ be the stations on a longest path P in MST. The first step is nearly identical to the line version: We direct all edges in MST towards s_1. Now consider the MST with root s_1, i.e., the arcs are now always directed from son to father. For notational convenience, we define a function $f : \{s_1, \ldots, s_l\} \to \mathbb{R}_+$ as follows. For a node s_i on P, consider the subtree T_i of MST rooted at s_i. We say there is *another edge at distance* r from s_i if there is a path P_r on stations $s_i, v_1, \ldots, v_{j-1}, v_j$ in T_i satisfying the following three conditions:

- $|P_r| \leq r$,
- $|P_r \setminus \{v_{j-1}, v_j\}| \geq r$,
- $v_j \notin \{s_1, \ldots, s_n\}$.

We now set $f(s_i) = r$ where $r \geq 0$ is the maximal r so that for every $0 \leq r' \leq r$, there is another edge at distance r' from s_i. Note that we have that $f(s_i) > 0$ iff station s_i has degree 3 or more in MST, i.e., it has at least two sons in T_i.

We are now ready to describe the second step of our generalized Combing algorithm. We apply the second step of Combing on the line to the path P in MST, with the following modification. Recall that for connecting station s_j we greedily chose station s_k with $k < j$ whose range came already closest to s_k. Now we do not necessarily raise the range of s_k to $d(s_k, s_j)$, but to

$$\max \{d(s_k, s_j)\} \cup \{f(s_{j'}) + d(s_k, s_{j'}) \mid k \leq j' < j\}. \qquad (*)$$

Then we repeat the loop as before until station s_l is reached.

Theorem 4.26. *Generalized Combing is a* 1.5*-approximation algorithm for metric* STRONG CONNECTIVITY.

Proof. We ensure overall strong connectivity by using expression $(*)$ for increasing the radii. In this way, all side paths in MST diverting from P are covered, and thus all stations.

The approximation ratio can be seen as follows. When expression $(*)$ is maximized by $d(s_k, s_j)$, the argument is identical to the line case. In the other cases, our increased radius does not only cover stations on P, but also stations along at least one other path diverting from P. This means by spending radius r, we provide connectivity along length $2r$ along MST (due to the $\triangle$-inequality), and we are in the same situation as in the other case, namely paying at most cost $3r$ on at least $2r$ units of $|MST|$, showing altogether approximation ratio 1.5. $\square$

It is not difficult to see that approximation factor 1.5 is tight for this metric generalization of Combing. A different, more careful treatment of side paths may yield a better performance. As the above actually quite easy procedure turned out to be quite involved to write down in detail, we refrained from constructing a more involved algorithm. Besides, the first step would have to be improving the analysis on the real line. On this behalf, however, we will need to find a better lower bound than MST. This alone would justify a more rigorous treatment of the Combing heuristic on the line, as a new lower bound may well fuel whole new approximations for these problems.

4.5 Overview and conclusion

In this chapter, we started out with the well-known important MST-heuristic and gave a tight analysis of its performance as a function of the instance size. Then we gave the first extensive analysis of two natural greedy algorithms for Connectivity which we called Kruskal and Prim due to their similarities to the two fundamental MST algorithms. We could show that both algorithms are 2-approximations, and that, despite their overcoming of the MST-heuristics tightness example, the factor 2 is tight already in many important settings by giving various tight examples. Still, for instance the case of Euclidean geometric instances is not yet resolved, and we conjecture that both greedy algorithms perform strictly better than 2 in the worst case. This would be a striking difference to the MST-heuristic, where the factor 2 is tight already on the Euclidean line. However, a proof would probably have to make extensive use of geometric properties, as we have shown that the factor 2 is tight on general metric instances. Furthermore, it has to make use of a better lower bound than the $|MST|$ lower bound, as we could show that there are instances on the Euclidean line where the cost of an optimal range assignment comes arbitrarily close to $2 \cdot |MST|$.

This is different for metric Strong Connectivity, and in the literature there already exists a 1.5-approximation algorithm, which also uses the $|MST|$ lower bound. This factor is tight already on very simple instances on the Euclidean line, where it also becomes immediately completely hopeless for geometric instances for any $\alpha > 1$. We proposed a new algorithm, called 'Combing', which is also a 1.5-approximation for metric instances, and is still a 2-approximation on the line for geometric or, more generally, linear instances. There are chances that its performance is actually better than 1.5, but for such a proof we would need to break through the limitations of the $|MST|$ lower bound.

At this point we would like to mention that there do exist algorithms with approximation ratio better than 2 for Connectivity. A group around Zelikovsky have shown a $(5/3 + \varepsilon)$-approximation scheme, and a more practical $(11/6)$-approximation algorithm in [ACM+03]. These two algorithms are very similar

to the two Steiner tree approximation algorithms of the same approximation ratios, the (11/6)-algorithms by Zelikovsky [Zel93], and the $(5/3+\varepsilon)$-approximation scheme by Prömel and Steger [PS00]. What is a striking parallel is that for the Steiner tree problem, a comparatively classic problem in combinatorial optimization, breaking through the $|MST|$ lower bound and proving a better approximation ratio than 2 has been a major open problem for decades, up until the above breakthrough work of Zelikovsky.

However, until now no algorithm is known with a better approximation ratio than 2 for Strong Connectivity. We could show that no Connectivity-algorithm can be used for such a result, so a better approximation would have to be purpose-built for Strong Connectivity. Also in this case, the $|MST|$ lower bound has to be improved upon, as there are geometric instances on the real line of optimal cost approaching $2|MST|$ for any $\alpha > 1$.

Bibliography

[ACM+03] E. Althaus, G. Călinescu, I. Măndoiu, S. Prasad, N. Tchervenski, and A. Zelikovsky. Power efficient range assignment in ad-hoc wireless networks. In *Proc. IEEE WCNC*, pages 1889–1894, 2003.

[ACP+04] C. Ambühl, A. E. F. Clementi, P. Penna, G. Rossi, and R. Silvestri. Energy consumption in radio networks: Selfish agents and rewarding mechanisms. *Theoretical Computer Science*, 343(1-2):27–41, 2004.

[Amb05] C. Ambühl. An optimal bound for the MST algorithm to compute energy efficient broadcast trees in wireless networks. In *Proceedings of 32nd International Colloquium on Automata, Languages and Programming (ICALP)*, volume 3580 of *Lecture Notes in Computer Science*, pages 1139–1150. Springer Verlag, 2005.

[APMS+99] G. Ausiello, M. Protasi, A. Marchetti-Spaccamela, G. Gambosi, P. Crescenzi, and V. Kann. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer Verlag, 1999.

[Aro98] S. Arora. Polynomial-time approximation schemes for Euclidean TSP and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998.

[AS98] S. Arora and M. Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998.

[Bak94] B. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994.

[BP89] M. W. Bern and P. E. Plassmann. The Steiner problem with edge lengths 1 and 2. *Information Processing Letters*, 32(4):171–176, 1989.

[CC03] M. Chlebík and J. Chlebíková. Inapproximability results for bounded variants of optimization problems. In *Proc. of 14th International*

Symposium on Fundamentals of Computation Theory (FCT 2003), LNCS 2751, pages 27–38. Springer, 2003. Also available as ECCC Report TR03-026.

[CCP+01a] A. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca. On the complexity of computing minimum energy. In *Proc. of 18th STACS*, LNCS 2010, pages 121–131, 2001.

[CCP+01b] A. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca. A worst-case analysis of an MST-based heuristic to construct energy-efficient broadcast trees in wireless networks. Technical Report 010, University of Rome "Tor Vergata", Math. Department, 2001.

[CHP+02] A. Clementi, G. Huiban, P. Penna, G. Rossi, and Y. Verhoeven. Some recent theoretical advances and open questions on energy consumption in ad-hoc wireless networks. In *Proc. 3rd Workshop on Approximation and Randomization Algorithms in Communication Networks (ARACNE)*, pages 23–38, 2002.

[CPS04] A. Clementi, P. Penna, and R. Silvestri. On the power assignment problem in radio networks. *Mobile Networks and Applications*, 9(2):125–140, 2004. Also available as ECCC Report TR00-054.

[Edm65] J. Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[Edm67] J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71:233–240, 1967.

[ESW96] P. Eades, C. Stirk, and S. Whitesides. The techniques of Kolmogorov and Bardzin for three-dimensional orthogonal graph drawings. *Information Processing Letters*, 60(2):97–103, 1996.

[Fei98] U. Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998.

[GJ77] M. Garey and D. Johnson. The rectilinear steiner tree problem is NP-complete. *SIAM Journal of Applied Mathematics*, 32(4):826–834, 1977.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, NY, USA, 1979.

[GJS76] M. Garey, D. Johnson, and L. Stockmeyer. Some simplified NP-complete problems. *Theoretical Computer Science*, 1:237–267, 1976.

[Hoc97] D. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems*. PWS Publishing, 1997.

[Kar72] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Communications*, pages 85–103. Plenum Press, 1972.

[KKKP00] L. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Theoretical Computer Science*, 243:289–305, 2000.

[Kru56] J. B. Kruskal. On the shortest spanning subtree of a graph and the travelling salesman problem. *Proceedings of the AMS*, 7:48–50, 1956.

[KV02] B. Korte and J. Vygen. *Combinatorial Optimization*. Springer, Berlin, Germany, 2nd edition, 2002.

[Mit99] J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: Part II – a simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM Journal of Computing*, 28(4):1298–1309, 1999.

[PL95] K. Pahlavan and A. Levesque. *Wireless Information Networks*. Wiley-Interscience, New York, 1995.

[Pri57] R. C. Prim. Shortest connection networks and some generalizations. *Bell Systems Technical Journal*, 36:1389–1401, 1957.

[PS00] H. J. Prömel and A. Steger. A new approximation algorithm for the steiner tree problem with performance ratio 5/3. *Journal of Algorithms*, 36:89–101, 2000.

[PS02] H. J. Prömel and A. Steger. *The Steiner Tree Problem*. Vieweg-Verlag, 2002.

[PY91] C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43:425–440, 1991.

[PY93] C. H. Papadimitriou and M. Yannakakis. The traveling salesman problem with distances one and two. *Mathematics of Operations Research*, 18:1–11, 1993.

[RS97] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and sub-constant error-probability PCP characterization of NP. In *Proc. 29th Ann. ACM Symp. on Theory of Comp.*, pages 475–484. ACM, 1997.

[RSL77] D. J. Rosenkrantz, R. E. Stearns, and P. M. Lewis II. An analysis of several heuristics for the Traveling Salesman Problem. *SIAM Journal on Computing*, 6(3):563–581, 1977.

[Tre04] L. Trevisan. Inapproximability of combinatorial optimization problems. Technical Report TR04-065, Electronic Colloquium on Computational Complexity (ECCC), 2004.

[Vaz03] V. Vazirani. *Approximation Algorithms*. Springer, 2nd edition, 2003.

[WaLF02] P.-J. Wan, G. Călinescu, X.-Y. Li, and O. Frieder. Minimum-energy broadcasting in static ad hoc wireless networks. *Wireless Networks*, 8:607–617, 2002.

[Zel93] A. Zelikovsky. An 11/6-approximation algorithm for the network Steiner problem. *Algorithmica*, 9:463–470, 1993.

Appendix A

Approximation algorithms in a nutshell

We assume the reader is familiar with the basic notions of combinatorial optimization. In this appendix, we want to give a short introduction to the theory of approximation algorithms and non-approximability. For more literature on this fascinating subject, the reader may want to study, among others, the book by Vazirani [Vaz03], the book by Ausiello *et al.* [APMS+99], the book edited by Hochbaum [Hoc97], or the survey by Trevisan [Tre04].

Recall that an instance I of an optimization problem P consists of a set of feasible solutions F together with a cost function c on F. The objective may be to find a feasible solution of maximal or minimal cost. As we only consider minimization problems in this thesis, we restrict ourselves to such problems here as well.

For an NP-hard optimization problem, we cannot hope for efficient exact algorithms, so it is worthwhile to study efficient *heuristics* in this case. A special class of heuristics are so-called *approximation algorithms*, which come with a certain guarantee of how far their solutions may differ from an optimal solution.

Definition A.1. Given a minimization problem P, and an algorithm $\mathcal{A}$. We say that $\mathcal{A}$ is a *ρ-approximation algorithm for P*, when we have that

$$\max\left\{\frac{c(\mathcal{A}(I))}{opt(I)}\right\} \leq \rho,$$

where $opt(I)$ is the cost of an optimal solution for instance I.

It is nowadays standard notation to call the class of optimization problems with a constant factor approximation APX. The usual notion of polynomial reducibility is not strong enough to preserve approximation properties in this class. In [PY91], the following type of reduction in this class has been defined.

Definition A.2 ([PY91]). Let P and P' be two minimization problems. We say that P *L-reduces to* P' if there are two polynomial-time algorithms f, g, and constants $\alpha, \beta > 0$ such that for each instance I of P:

- Algorithm f produces an instance $I = f(I)$ of P, such that the optima of I and I, $opt(I)$ and $opt(I)$, respectively, satisfy $OPT(I) \leq \alpha \cdot OPT(I')$, and
- Given any solution of I with cost c, algorithm g produces a solution of I with cost c such that $c - OPT(I) \leq \beta(c - OPT(I'))$.

Two immediate yet crucial observations are:

Lemma A.3 ([PY91]). *L-reductions compose.*

Lemma A.4 ([PY91]). *If P L-reduces to P', and there is a polynomial-time $(1 + 1/(\alpha\beta\gamma))$-approximation algorithm for P', then there is a polynomial-time $(1 + 1/\gamma)$-approximation algorithm for P.*

We say that a problem in APX is *APX-complete* if every problem in APX L-reduces to it.

An optimization problem has a polynomial time approximation scheme (or PTAS) if for every constant $\varepsilon > 0$, there is a polynomial time $(1+\varepsilon)$-approximation algorithm. (Note the running time may well be exponential in $1/\varepsilon$.) The class of problems allowing a PTAS thus is a subclass of APX.

Note that when a problem P' allows a PTAS, and P L-reduces to P', then P allows a PTAS too. Via the famous PCP-theorem [AS98] it could be proved that there are APX-complete problems, such as Max 3-SAT that do not allow a PTAS unless P=NP. Together with the notion of L-reducibility, this implies the following Theorem.

Theorem A.5. *There cannot be a PTAS for any APX-hard problem unless P=NP.*

This means that showing APX-hardness of a problem makes the search of a PTAS for this problem as obsolete as trying to design an efficient exact algorithm for an NP-hard problem.

In chapter 3, we prove APX-hardness for several problems. The reductions used are all L-reductions from known APX-hard problems.

Erklärung

Ich versichere, dass ich die von mir vorgelegte Dissertation selbständig angefertigt, die benutzten Quellen und Hilfsmittel vollständig angegeben und die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken im Wortlaut oder dem Sinn nach entnommen sind, in jedem Einzelfall als Entlehnung kenntlich gemacht habe, dass diese Dissertation noch keiner anderen Fakultät oder Universität zur Prüfung vorgelegen hat, dass sie – abgesehen von unten angegebenen Teilpublikationen – noch nicht veröffentlicht worden ist sowie, dass ich eine solche Veröffentlichung vor Abschluss des Promotionsverfahrens nicht vornehmen werde. Die Bestimmungen der Promotionsordnung sind mir bekannt. Die von mir vorgelegte Dissertation ist von Prof. Dr. R. Schrader betreut worden.

Teilpublikation:

- B. Fuchs. On the hardness of range assignment problems. *Networks*, to appear. An extended abstract appeared in *Proc. 6th Italian Conference on Algorithms and Complexity*, LNCS 3998, pages 127–138, 2006. Also available as ECCC Report TR05-113 at http://eccc.hpi-web.de/eccc/.